EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION USING NEURAL-BASED ESTIMATION OF DISTRIBUTION ALGORITHMS

SHIM VUI ANN

NATIONAL UNIVERSITY OF SINGAPORE

2012

EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION USING NEURAL-BASED ESTIMATION OF DISTRIBUTION ALGORITHMS

SHIM VUI ANN

B.Eng (Hons., 1st Class), UTM

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2012

Acknowledgments

The accomplishment of this thesis had to be the ensemble of many causes. First and foremost, I wish to express my great thanks to my Ph.D. supervisor, Associate Professor Tan Kay Chen, for introducing me to the wonderful research world of computational intelligence. His kindness has provided a pleasant research environment; his professional guidance has kept me in the correct research track during the course of my four years of research; and his motivation and advice have inspired my research.

My great thanks also goes to my seniors as well as other lab buddies who have shared their experience and helped me from time to time. The diverse background and behaviour among my buddies have made my university life memorable and enjoyable: Chi Keong for being the big senior in the lab who would occasionally drop by to visit and provide guidance, Brian for being the cheer leader, Han Yang for providing incredible philosophical views, Chun Yew for demonstrating the steady and smart way of learning, Chin Hiong for sharing his professional skills, Jun Yong for accompanying me in the intermediate course of my studies, Calvin for sharing his working experiences, Tung for teaching me the computer skills, HuJun and YuQiang for being the replacements, and Sen Bong for accompanying me in the last year of my Ph.D. studies. I would also like to express my gratitude to the lab officers, HengWei and Sara, for their continuous assistance in the Control and Simulation lab.

Last but not least, I would like to express my deep seated appreciation to my family for their selfless love and care. This thesis would not be possible without the ensemble of these causes.

Contents

Summary

Multi-objective optimization is widely found in many fields, such as logistics, economics, engineering, bioinformatics, finance, or any problems involving two or more conflicting objectives that need to be optimized simultaneously. The synergy of probabilistic graphical approaches in evolutionary computation, commonly known as estimation of distribution algorithms (EDAs), may enhance the iterative search process when probability distributions and interrelationships of the archived data have been learnt, modelled, and used in the reproduction. The primary aim of this thesis is to develop a novel neural-based EDA in the context of multi-objective optimization and to implement the algorithm to solve problems with vastly different characteristics and representation schemes.

Firstly, a novel neural-based EDA via restricted Boltzmann machine (REDA) is devised. The restricted Boltzmann machine (RBM) is used as a modelling paradigm that learns the probability distribution of promising solutions as well as the correlated relationships between the decision variables of a multi-objective optimization problem. The probabilistic model of the selected solutions is derived from the synaptic weights and biases of RBM. Subsequently, a set of offspring are created by sampling the constructed probabilistic model. The experimental results indicate that REDA has superior optimization performance in high-dimensional and many-objective problems. Next, the learning abilities of REDA as well as its behaviours in the perspective of evolution are investigated. The findings of the investigation inspire the design of a novel energy-based sampling mechanism which is able to speed up the convergence rate and improve the optimization performance in both static and epistatic test problems.

REDA is also extended to study the multi-objective optimization problems in noisy environments, in which the objective functions are influenced by a normally distributed noise. An enhancement operator, which tunes the constructed probabilistic model so that it is less affected by the solutions with large selection errors, is designed. A particle swarm optimization algo-

rithm is hybridized with REDA in order to enhance its exploration ability. The results reveal that the hybrid REDA is more robust than the algorithms with genetic operators in all levels of noise. Moreover, the scalability study indicates that REDA yields better convergence in high-dimensional problems.

The binary-number representation of REDA is then modified into integer-number representation to study the classical multi-objective travelling salesman problem. Two problem-specific operators, namely permutation refinement and heuristic local exploitation operators are devised. The experimental studies show that REDA has a faster and better convergence but poor solution diversity. Thus, REDA is hybridized with a genetic algorithm, in an alternative manner, in order to enhance its ability in generating a set of diverse solutions. The hybridization between REDA and GA creates a synergy that ameliorates the limitation of both algorithms. Next, an advance study of REDA in solving the multi-objective multiple travelling salesman problem (MmTSP) is conducted. A formulation of the MmTSP, which aims to minimize the total travelling cost of all salesmen and balancing of the workloads among all salesmen, is proposed. REDA is developed in the decomposition-based framework of multi-objective optimization to solve the formulated problem. The simulation results reveal that the proposed algorithm successes in generating a set of diverse solutions with good proximity results.

Finally, REDA is combined with a genetic algorithm and a differential evolution in an adaptive manner. The adaptive algorithm is then hybridized with the evolutionary gradient search. The hybrid adaptive algorithm is constructed in both the domination-based and decomposition-based frameworks of multi-objective optimization. Even through only three evolutionary algorithms (EAs) are considered in this thesis, the proposed adaptive mechanism is a general approach which can combine any number of search algorithms. The constructed algorithms are tested under 38 global continuous test problems. The algorithms are successful in generating a set of promising approximate Pareto optimal solutions in most of the test problems.

Lists of publications

The publications that was published, accepted, and submitted during the course of my research are listed as follows.

Journals

1. V. A. Shim, K. C. Tan, C. Y. Cheong, and J. Y. Chia, "Enhancing the Scalability of Multi-objective Optimization via a Neural-based Estimation of Distribution Algorithm", *Information Sciences*, submitted.

2. V. A. Shim, K. C. Tan, and C. Y. Cheong, "An Energy-based Sampling Technique for Multi-objective Restricted Boltzmann Machine", *IEEE Transactions on Evolutionary Computation*, in revision.

3. V. A. Shim, K. C. Tan, J. Y. Chia, and A. Al. Mamun, "Multi-objective Optimization with Estimation of Distribution Algorithm in a Noisy Environment", *Evolutionary Computation*, accepted, 2012.

4. V. A. Shim, K. C. Tan, J. Y. Chia, and J. K. Chong, "Evolutionary Algorithms for Solving Multi-objective Travelling Salesman Problem", *Flexible Services and Manufacturing Journal*, vol. 23, no. 2, pp. 207-241, 2011.

5. V. A. Shim, K. C. Tan, and C. Y. Cheong, "A Hybrid Estimation of Distribution Algorithm with Decomposition for Solving the Multi-objective Multiple Traveling Salesman Problem". *IEEE Transactions on Systems, Man, and Cybernetic: Part C*, vol. 42, no. 5, pp. 682-691, 2012.

6. J. Y. Chia, C. K. Goh, K. C. Tan, and V. A. Shim, "Memetic informed evolutionary optimization via data mining". *Memetic Computing*, vol. 3, no. 2, pp. 73-87, 2011.

7. J. Y. Chia, C. K. Goh, V. A. Shim, and K. C. Tan, "A data mining approach to evolutionary optimisation of noisy multi-objective problems". *International Journal of Systems Science*, vol. 43, no. 7, pp. 1217-1247, 2012.

Conferences

1. H. J. Tang, V. A. Shim, K. C. Tan, and J. Y. Chia, "Restricted Boltzmann Machine Based Algorithm for Multi-objective Optimization", *in IEEE Congress on Evolutionary Computation*, pp. 3958-3965, 2010.

2. V. A. Shim, K. C. Tan, and J. Y. Chia, "An Investigation on Sampling Technique for Multi-objective Restricted Boltzmann Machine", *in IEEE Congress on Evolutionary Computation*, pp. 1081-1088, 2010.

3. V. A. Shim, K. C. Tan, and J. Y. hia, "Probabilistic based Evolutionary Optimizers in Bi-objective Traveling Salesman Problem", *in Eighth International Conference on Simulated Evolution and Learning*, pp. 588-592, 2010.

4. V. A. Shim, K. C. Tan, and K. K. Tan, "A Hybrid Estimation of Distribution Algorithm for Solving the Multi-objective Multiple Traveling Salesman Problem", *in IEEE Congress on Evolutionary Computation*, pp. 771-778, 2012.

5. V. A. Shim, K. C. Tan, and K. K. Tan, "A Hybrid Adaptive Evolutionary Algorithm in the Domination-based and Decomposition-based Frameworks of Multi-objective Optimization", *in IEEE Congress on Evolutionary Computation*, pp. 1142-1149, 2012.

Book Chapter

1. V. A. Shim and K. C. Tan, "Probabilistic Graphical Approaches for Learning, Modeling, and Sampling in Evolutionary Multi-objective Optimization", *in J. Liu et al. (Eds.): IEEEWCCI 2012, LNCS 7311, Springer, Heidelberg*, pp. 122-144, 2012.

List of Tables

List of Figures

Chapter 1

Introduction

Many real-world problems involve the simultaneous optimization of several conflicting objectives that are difficult, if not impossible, to solve without the aid of powerful optimization algorithms. For example, when travelling from workplace to home, a commuter may consider the cheapest and most convenient means of transportation. The cheapest may not be the most convenient, and therefore the two objectives are conflicting. This kind of problem is commonly known as a multi-objective optimization problem (MOP). MOP is a difficult optimization problem because no one solution is optimal for all objectives. Therefore, in order to solve an MOP, search methods employed must be capable of finding a number of alternative solutions representing the tradeoff between the various conflicting objectives. In addition to finding a set of tradeoff solutions, the search methods may encounter other difficulties of MOPs, including complex, non-linear, non-differentiable, constrained, and high-dimensional search space. Due to these difficulties, most deterministic optimization techniques fail to obtain reasonable solutions in the limited computational resource. In addressing these issues, stochastic search techniques appear to be more suitable than deterministic optimization techniques.

In the literature, many simple MOPs have been effectively solved by using evolutionary algorithms (EAs). EAs are stochastic and population-based approaches inspired from biological evolution [1, 2], and they consist of several characteristics. First, EAs sample multiple candidate

solutions in a single simulation run. Second, EAs apply the concept of survival-of-the-fittest to maintain the candidate solutions who have been found. Third, EAs implement stochastic recombination operators inspired from biological evolution to explore the search space. Due to these characteristics, EAs have been successfully implemented to solve many application problems. Some examples of the implementation of EAs include optimization of grid task scheduling with multi-QoS constraint [3], reservoir system [4], economic power dispatch [5], and pump scheduling [6], just to name a few. Nonetheless, the stochastic recombination operators in EAs may disrupt the building of strong schemas and the movement towards the optimal is extremely difficult to predict [7].

In order to overcome the aforementioned limitations of EAs, the estimation of distribution algorithm (EDA), which is motivated by the idea of exploiting the probability information of promising solutions, has been regarded as a new computing paradigm in the field of evolutionary computation [7, 8]. In contrast to EAs, EDA does not implement any stochastic recombination operators to generate new solutions. Instead, the new solutions are produced by building a representative probabilistic model of the maintained tradeoff solutions, and subsequently sampling the constructed probabilistic model. The probabilistic model can be built by considering the linkage information of solutions in the decision space. The model is used to predict global movement of the solutions during the search process. With regard to modelling issues, many modelling approaches, including statistical methods, probability approaches, graphical models, and neural-based mechanisms, can be implemented. Among these modelling approaches, the neural-based mechanism, specifically the restricted Boltzmann machine (RBM), is one of the promising methods due to the learning behaviour of the network. Furthermore, RBM is able to capture the interdependencies of the parameters, is easy to implement, and is easily adapted to suit the framework of EDAs without substantial modification to the architecture of the network. With these advantages, the use of the probabilistic information modelled by RBM would help in predicting the movements in the search space, which may lead the search to approach optima.

1.1 Multi-objective Optimization

Multi-objective optimization problems (MOPs) are widely found in many application fields, such as scheduling, finance, engineering, data mining, and bioinformatics, among others. The principles behind multi-objective optimization have been studied over the past decades. This section introduces the basic concepts and principles of multi-objective optimization.

1.1.1 Basic Concepts

A multi-objective optimization problem (MOP), which involves the simultaneous optimization of several conflicting objectives to satisfy problem constraints, is a difficult and complex problem. Mathematically, an MOP can be formulated, in the minimization case, as follows:

Minimize:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x})) \tag{1.1}$$

subject to:

$$\mathbf{g}(\mathbf{x}) \leq 0$$

$$\mathbf{h}(\mathbf{x}) = 0$$

where $\mathbf{f}(\mathbf{x})$ is the set of objective functions, $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^m, \mathbb{R}^m$ is the objective space, m is the number of objective functions, $\mathbf{x} = (x_1, x_2, ..., x_n)$ is the decision vector, $\mathbf{x} \in \mathbb{R}^n$, $\mathbb{R}^n$ is the decision space, n is the number of decision variables, $\mathbf{g}$ is the set of inequality constraints, and $\mathbf{h}$ is the set of equality constraints.

In an MOP, no single point is an optimal solution. Instead, the optimal solution is a set of non-dominated solutions, which represents the tradeoff between the multiple objectives. In other words, the improvement in one objective can only be achieved with the detriment in at least one other objective. In this case, the fitness assignment to each solution in the evolutionary

framework is considered as an important feature for the assurance of the survival of fitter and less crowded solutions to the next generation.

1.1.2 Pareto Optimality and Pareto Dominance

In the literature, the concepts of Pareto optimality and Pareto dominance have been widely used to describe the optimal solutions for an MOP and to define criteria for solution comparison.

Let $\mathbf{a} = (a_1, ..., a_n)$ and $\mathbf{b} = (b_1, ..., b_n)$ represent two decision vectors of solutions that consist of n decision variables. In the context of Pareto optimality, three relations between the two solutions can be defined [9–11]. These relationships, in the minimization case, are listed below:

1. Strong dominance: $\mathbf{a}$ is said to strongly dominate $\mathbf{b}$ ($\mathbf{a} \prec \mathbf{b}$) if and only if

$$f_i(\mathbf{a}) < f_i(\mathbf{b}) \; \forall i \in \{1, 2, ..., m\} \tag{1.2}$$

2. Weak dominance: $\mathbf{a}$ is said to weakly dominate $\mathbf{b}$ ($\mathbf{a} \preceq \mathbf{b}$) if and only if

$$f_i(\mathbf{a}) \leq f_i(\mathbf{b}) \text{ for } i \in \{1, 2, ..., m\} \text{ and } \exists \; f_i(\mathbf{a}) < f_i(\mathbf{b}) \text{ for at least one } i \tag{1.3}$$

3. Incomparable: $\mathbf{a}$ and $\mathbf{b}$ are incomparable ($\mathbf{a} \sim \mathbf{b}$) if and only if

$$\exists i \in \{1, 2, ..., m\} : f_i(\mathbf{a}) > f_i(\mathbf{b}) \text{ and } \exists j \in \{1, 2, ..., m\} : f_j(\mathbf{a}) < f_j(\mathbf{b}) \tag{1.4}$$

The dominance relationships between solutions for a two-objective example are further illustrated in Figure 1.1. Let solution **U** be the reference solution and the dominance relations are highlighted in different shaded regions (dark grey, light grey, and white). Solution **U** strongly dominates solutions located in the dark grey region because solution **U** is better in both objectives. On the other hand, solution **U** is strongly dominated by solutions in the white region since these solutions have better objective values than solution **U**. For solutions that lie in the boundaries of the shaded regions, they share the same objective value in one of the objectives as solution **U**,

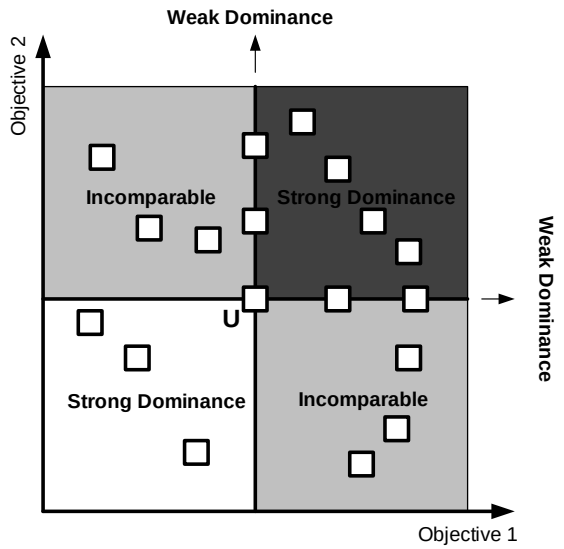


Figure 1.1: The concept of Pareto dominance

but solution **U** has a better objective value in another objective. Thus, these solutions are weakly dominated by solution **U**. For solutions located in the grey regions, they are superior in one of the objective functions, while are inferior in another objective function compared to solution **U**. Thus, these solutions are incomparable to solution **U**.

A decision vector $\mathbf{x}^* \in \mathbb{R}^n$ is said to be non-dominated if and only if $\nexists \mathbf{b} \in \mathbb{R}^n : \mathbf{b} \prec \mathbf{x}^*$ and $\mathbf{x}^*$ is a Pareto optimal solution. The set of all Pareto optimal decision vectors is called the Pareto optimal set (PS) and the corresponding objective vectors form the Pareto optimal front (PF) [1].

The Pareto optimal front of an MOP is illustrated in Figure 1.2. In the rest of this thesis, 'weakly dominate' and 'strongly dominate' are simply termed as 'dominate'. In the figure, F1 is the first objective and F2 is the second objective. Solutions A, B, C, and D are mutually non-dominated and solutions B and C dominate solution E. A set of non-dominated solutions (A, B, C, and D) will form the Pareto optimal front.

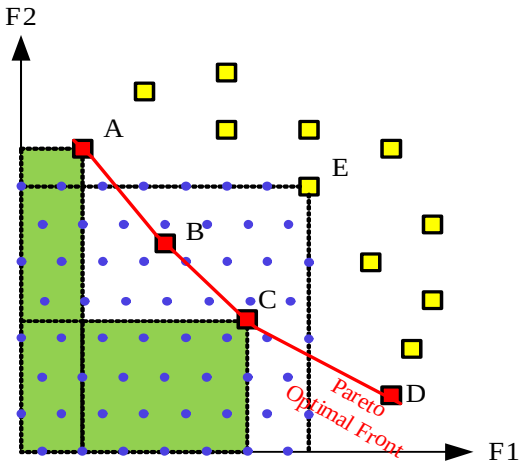


Figure 1.2: Illustration of Pareto optimal front

1.1.3 Goals of Multi-objective Optimization

In performing a multi-objective optimization, there is no guarantee that an algorithm, especially a heuristic-based algorithm, can obtain a set of ideal optimal solutions. Due to the difficulties of real-world optimization problems, the aim of the multi-objective optimization is to find an approximate set of solutions that is as close to the Pareto optimal front as possible. Thus, it is necessary to define a set of criteria to describe how good the generated set of solutions is. These criteria [9, 12] are presented as follows:

1. **Proximity:** Determine how close the obtained solutions are to the Pareto optimal front.

2. **Diversity:** Determine how well the obtained solutions are distributed along the Pareto optimal front.

3. **Spacing:** Determine how evenly distributed the obtained solutions are along the Pareto optimal front.

4. **Number of non-dominated solutions:** Determine the number of non-dominated solutions generated by an algorithm.

The proximity, which defines the distance between the obtained solutions and the optimal solutions, is the main criterion of all optimization problems. Meanwhile, the other three criteria are unique to multi-objective optimization since the optimal solutions of an MOP are a set of tradeoff solutions. For diversity and spacing, these criteria describe how the obtained solutions are distributed in the optimal space. The diversity defines how well is the coverage of the obtained solutions in the optimal space while the spacing defines how evenly distributed are the multiple solutions in the optimal space. A set of diverse solutions with uniform distribution are crucial in MOPs as they provide multiple choices to decision makers before the final choice is made. The last criterion determines how many non-dominated solutions are generated by an algorithm.

1.1.4 The Frameworks of Multi-objective Optimization

Over the past three decades, several frameworks of multi-objective optimization have been proposed to tackle MOPs. The framework of multi-objective optimization refers to the approach that an algorithm takes to handle multiple conflicting objectives. In [13], the author classified the frameworks into three main categories. First, an MOP can be decomposed into a single-objective optimization problem by combining the multiple conflicting objectives into a single-objective function. Second, an MOP can be solved by optimizing one objective at a time while considers other objectives as constraints. Third, an MOP can be solved by optimizing all objectives simultaneously. The concept of Pareto dominance is particularly useful in this approach.

The above classification is outdated and not covers many other frameworks of multi-objective optimization. In this thesis, we present a more general classification of the frameworks of multi-objective optimization as follows:

1. **Preference-based Framework:** The basic idea of this framework is to aggregate the multiple conflicting objectives of MOPs into a single-objective optimization problem or to use preference knowledge of the problems so that the optimizers can focus on optimizing certain objectives. Then, a common EA for solving single-objective optimization problems is directly

applied to solve the aggregated function. The first approach classified in [13] is a subset of this framework. However, this framework suffers two major limitations. First, only one approximate optimal solution can be obtained in a simulation run. Second, it is necessary to specify a weight vector or a preference of managers for the purpose of aggregation.

2. **Domination-based Framework:** In this approach, an MOP is solved by optimizing all objectives simultaneously. Fitness assignment to each solution in this framework is an important feature for the assurance of the survival of fitter solutions to the next generation. Pareto dominance is particularly useful in defining the superiority of each solution with regards to the whole solution set. This approach is effective in generating a set of tradeoff solutions. For this reason, it has gained extensive attention from the research community. This framework is identical to the third approach classified in [13]. However, a major drawback of this framework is that the selective pressure is weakened with the increase in the number of objective functions. Furthermore, it is necessary to specify a diversity preservation scheme in order to maintain a set of diverse solutions.

3. **Decomposition-based Framework:** This framework decomposes an MOP into several subproblems where a subproblem is constructed by using any aggregation-based methods. After that, all the subproblems are optimized concurrently. The selective pressure problem as faced by the domination-based framework does not exist in this framework since the fitness of a solution solely depends on the aggregated objective value. Moreover, it is not necessary to specify a diversity preservation scheme, which is required in the domination-based framework, since the diversity can be preserved by using the predefined uniformly distributed weight vectors. This framework has gained increasing attention from the research community recently.

1.2 Evolutionary Algorithms in Multi-objective Optimization

Many optimization algorithms can be used to deal with MOPs. In the literature, evolutionary algorithms (EAs) are one of the most popular approaches to solve MOPs. Therefore, this thesis focuses on the implementation of EAs to solve MOPs. In this section, the general concept of a typical EA and its basic process flow in multi-objective optimization are presented.

1.2.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) [1] are computing paradigms, which mimic the nature of evolution in driving the search towards optimality. Survival-of-the-fittest and genetic recombination are the main concepts for the success of EAs. EAs have been recognized to be a general purpose optimization tool due to two main reasons. First, EAs do not take into account any background knowledge of problems when performing optimization. Second, EAs do not require any gradient or directional information in exploring the search space. Instead, the heuristic nature of the genetic searches in EAs allow them to efficiently perform the searching task in any fitness landscape.

The basic idea of a typical EA is illustrated in Figure 1.3. An EA begins by randomly initializing a set of solutions to form an initial population. The population is evolved and a set of fitter solutions are preserved over the evolutionary process. In the evolutionary process, the fitness of the solutions is evaluated. Subsequently, only a set of fitter solutions are selected to undergo genetic operations. The selected solutions are identified as parent solutions who will mate among themselves through the crossover operation in order to produce offspring. A detailed description of the selection operators can be found in [14, 15]. In the crossover operation, two solutions are randomly selected from the mating pool (parent population). The alleles of the solutions are exchanged so that the child solutions will possess the characteristics of both parent solutions. The proper implementation of the crossover operators (e.g. single-point crossover, multi-point crossover, etc.) is one of the key successes of EAs in exploring the search space [16].

Some of the generated offspring are then exposed to mutation operation by means of randomly perturbing some alleles. The function of mutation operators (e.g. swap mutation, bit-flip mutation, polynomial mutation, etc.) is to prevent the search from converging to local optima. The generated child solutions or offspring and the parent solutions are then stored in an archive. The core of the evolutionary process, which is the concept of the survival-of-the-fittest [17], follows. Through this concept, the fitter solutions between the parent and offspring, which are marked by the fitness of the solutions, are selected to form a new population that will undergo evolution in the next generation. The process continues until a stopping criterion is satisfied. The best solution found in the evolutionary process is considered as the approximated optimal solution.

Begin
Initialization: Randomly initialize a population
Do While ("Stopping Criterion is not satisfied")
Evaluation: Calculate the fitness of each solution in the population
Selection: Select a set of parent solutions
Variation Operators: Perform crossover and mutation to the parent solutions to create offspring
Archiving: Store the promising solutions (parent and offspring) in an archive
Elitism: Form a new population by selecting solutions from the archive
End Do
End

Figure 1.3: Pseudo-code of a typical EA

1.2.2 Multi-objective Evolutionary Algorithms

EAs, which are general optimization algorithms, are naturally suitable to solve MOPs. This is because the population-based approach of EAs allows approximating a set of tradeoff solutions in a single simulation run. Furthermore, the heuristic search of EAs enable solving a wide variety of problems in which the characteristics of the problems are unknown. The implementation of EAs in the framework of multi-objective optimization is commonly known as multi-objective evolutionary algorithms (MOEAs) [1, 2, 18].

An MOEA shares a similar process flow as a typical EA as illustrated in Figure 1.3. Two

main differences are: fitness assignment and output solutions. In MOEAs, the fitness of solutions in a population cannot be directly assigned to be the objective value of the solutions as performed in a typical EA, due to the involvement of multiple conflicting objectives that have to be simultaneously optimized. In this case, the fitness assignment to each solution in an evolutionary process is considered as an important feature for the assurance of the survival of fitter and less crowded solutions to the next generation. The simplest way to assign a fitness to the solutions is through the aggregation approach as performed in the preference-based and decomposition-based frameworks of multi-objective optimization. Another approach is to find the domination correlations between the solutions in a population as performed in the domination-based framework of multi-objective optimization.

In MOEAs, multiple tradeoff solutions are the output solutions from an evolutionary process instead of a single best solution as output from a typical EA. The need of MOEAs in maintaining a diverse set of promising solutions throughout the evolutionary process poses another level of challenge to the optimizers.

1.3 Estimation of Distribution Algorithms in Multi-objective Optimization

Over the past few decades, the implementation of EAs in solving MOPs has gained remarkable attention from the research community [2, 18, 19]. Nonetheless, stochastic recombination in standard EAs (genetic algorithm (GA) in particular) may disrupt the building of strong schemas. The movement towards the optima is, thus, extremely difficult to predict [7, 20]. It is necessary to specify the settings of several parameters (e.g. crossover and mutation rates) for optimization. Due to these reasons, the estimation of distribution algorithms (EDAs), motivated by the idea of exploiting the probability information of promising solutions, has been introduced as a new computing paradigm in the field of evolutionary computation.

EDAs are similar to GA in the sense that they also mimic the principle of biological evolution, which is the survival-of-the-fittest, to guide the search. However, the primary difference between EDAs and GAs is that there is no implementation of genetic operators (crossover and mutation) in EDAs. Instead, the reproduction is carried out by the building of a representative probabilistic model of candidate solutions. Subsequently, child solutions are generated through the sampling of the constructed model. This reproduction strategy can prevent the disruption of the strong schema during the evolutionary process. The probabilistic-based approach in EDAs provides a strong search behaviour through the consideration of the global probability distribution and linkage information (the probability of certain genes to be inherited by others) of the candidate solutions in the decision space. The discovered knowledge (probability distribution and linkage information) of the data is used to predict the location of the optimal solution or the favourable movement in the search space [7] or the pattern of the Pareto front in multi-objective case [21]. By using the discovered correlations of the parameters of a cost function, the search can be regulated to follow the correlated patterns when generating an offspring solution.

Begin
Initialization: Randomly initialize a population
Do While ("Stopping Criterion is not satisfied")
Evaluation: Calculate the fitness of each solution in the population
Selection: Select a set of parent solutions
Modeling: Build a probabilistic model of the parent solutions
Sampling: Generate a set of offspring by sampling from the probabilistic model
Archiving: Store the promising solutions (parent and offspring) in an archive
Elitism: Form a new population by selecting solutions from the archive
End Do
End

Figure 1.4: Pseudo-code of a typical EDA

The basic process flow of a typical EDA is illustrated in Figure 1.4. In the figure, it can be observed that GAs and EDAs share a common process flow. The only difference is that EDAs construct a probabilistic model to represent the probability distribution of the parent solutions and subsequently sample the model to generate offspring. This is the difference from GAs in

which the offspring are generated through the genetic recombination and mutation of the parent solutions.

Due to the success of EDAs in single-objective optimization, the implementation of EDAs for multi-objective optimization has been gaining research interest. It is of interest to note that EDAs can suit any framework of multi-objective optimization, which have been developed over the past few decades. This is because a typical EDA shares a common algorithmic flow as a typical EA. Thus, the fitness assignment approach and the diversity preservation mechanism in the aforementioned frameworks of multi-objective optimization can be directly employed by EDAs.

1.4 Objectives

Even though many attempts have been devoted to developing new algorithms for MOPs, the optimization performance of those algorithms in complex MOPs is still far from achievable results. The research gaps for the current study on MOPs are summarized below:

1. EAs, particularly GAs, have been extensively employed to solve MOPs. GAs implement stochastic recombination to generate new solutions. However, the stochastic recombination in GAs may disrupt the building of strong schemas of a population and the movement towards optima is extremely difficult to predict. Furthermore, it is necessary to specify the settings of certain parameters that govern the evolutionary process [7].

2. Modelling and sampling are two main issues in EDAs. A number of endeavours have been devoted to studying the modelling issue of EDAs. However, the study of the sampling issue of EDAs does not receive as much interest as the modelling study.

3. In real-world problems, some objective functions of MOPs are subject to uncertainty, which may be caused by the noise of input sensors, error in approximation, or unpredictable environment changes such as ambient change. The uncertain objective values of an MOP pose another

level of difficulty in optimization. Several attempts have been carried out to study the effects of noise in MOEAs. However, none has implemented multi-objective EDAs (MOEDAs) to study MOPs in noisy environments.

4. Several attempts have been carried out to study the performance of MOEDAs in solving discrete-valued and real-valued MOPs. However, none has studied the performance of MOEDAs in permutation-based MOPs such as scheduling problems.

5. There are three different frameworks to solve MOPs, namely preference-based, domination-based, and decomposition-based frameworks. The latter two frameworks have gained considerable attention due to their effective optimization performance. However, detailed investigations on their optimization performance on complex MOPs have yet to be conducted. Furthermore, none has studied the performance of MOEDAs under the decomposition-based framework of multi-objective optimization.

6. Under the framework of evolutionary paradigms, many variations of EAs have been designed. Each of the algorithms performs well in certain cases, and none of them dominate the others. Using an ensemble of multiple optimizers is another approach to complement the limitation of each algorithm. However, none has studied the issue using ensembles for MOEDAs.

The main aim of this study is to propose an algorithm that can solve a variety of MOPs effectively. The specific objectives of this research are:

1. To develop an algorithm for MOPs using restricted Boltzmann machine (RBM) based estimation of distribution algorithm (REDA).

2. To understand the behaviours of REDA in the evolutionary process and to study the sampling issue of REDA.

3. To study the optimization performance of REDA in noisy MOPs.

4. To adapt REDA for solving permutation-based problems, specifically the multi-objective travelling salesman problem (MTSP) and the multi-objective multiple travelling salesman problem (MmTSP).

5. To study the optimization performance of REDA under the domination-based and decomposition-based frameworks of multi-objective optimization.

6. To ensemble REDA with other global and local optimizers for solving MOPs with vastly different characteristics.

The proposed algorithm, REDA, which constructs the probabilistic model of the promising solutions and uses it to guide the search, may effectively search over the search space in some MOPs. This is because RBM is able to capture the inherent correlation information between the decision variables, and this information can be used to predict the global movement during the search process. The neural-based mechanism employed in REDA may provide a level of flexibility when constructing the probabilistic model of the promising solutions. This feature provides a platform to investigate the suitability of the probabilistic model in an evolutionary process. The implementation of REDA in noisy MOPs may provide interesting results and observations on the performance of MOEDAs in uncertain environments. For the implementation of REDA in scheduling problems, the optimization performance and behaviours of MOEDAs in general or REDA in particular in permutation-based MOPs can be investigated. This may contribute to a better understanding of the information mined in scheduling problems by MOEDAs. Since REDA is developed in the domination-based and decomposition-based frameworks of multi-objective optimization, the strengths and weaknesses of both frameworks in a variety of MOPs may be explored and understood. When REDA is hybridized with other optimizers, the hybrid algorithms should able to solve a variety of MOPs effectively.

This thesis focuses on the implementation of EAs to study MOPs. Specifically, EDAs are the main algorithms, GAs, differential evolution (DE), and particle swarm optimizer (PSO) are

the side algorithms. Other EAs including evolutionary strategy, genetic programming, and ant colony optimizer are not considered in this thesis. For the frameworks of multi-objective optimization, only the domination-based and decomposition-based frameworks are considered while the preference-based framework is not explored. This is because the previous two frameworks are more commonly used and are able to generate a set of tradeoff solutions in a single simulation run. Since there are many variances of MOPs, it is impossible to consider all of them. Thus, the MOPs considered in this thesis are limited to 31 benchmark global continuous test problems and two scheduling test problems. Combinatorial binary MOPs, constrained MOPs, and other real-world optimization problems are beyond the scope of this thesis.

1.5 Contributions

In this thesis, a neural-based estimation of distribution algorithm for solving a variety of multi-objective optimization problems has been devised. The algorithmic designs, implementations, experiments, analyses, and results are detailed. The itemized contributions of this research are listed below:

1. A restricted Boltzmann machine-based estimation of distribution algorithm (REDA) has been designed. This marks the possibilities of the synergy between evolutionary algorithms and neural networks for solving multi-objective optimization problems. This contribution is realized in chapter 3.

2. The behaviours of REDA in the evolutionary process have been extensively studied. This study provides a better understanding of the training, modelling, and sampling issues of an RBM in the perspective of evolution. This study also motivates the proposal of an energy-based sampling mechanism of REDA. The energy-based sampling mechanism successes in enhancing the search capability of REDA. This contribution is realized in chapter 4.

3. A first attempt to implement MOEDAs for solving MOPs in noisy environments has been

carried out in this thesis. This attempt provides a deeper understanding of the behaviours of MOEDAs in uncertain environments. This contribution is realized in chapter 5.

4. A first attempt to adapt MOEDAs for solving the multi-objective travelling salesman problem (MTSP) and the multi-objective multiple travelling salesman problem (MmTSP) has been implemented in this thesis. An adaptation approach of MOEDAs for solving permutation-based combinatorial optimization problems has also been suggested. This contribution is realized in chapters 6 and 7.

5. Extensive studies of the optimization performance of MOEDAs under the domination-based and decomposition-based frameworks of multi-objective optimization have been conducted in this research. These studies provide a better understanding of the strengths and weaknesses of both frameworks through experimental results comparison. This contribution is realized in chapters 7 and 8.

6. Several hybrid and memetic approaches for MOEDAs have been studied in this research. These studies have experimentally proved that the hybrid and memetic approaches are promising techniques in enhancing the optimization performance of MOEDAs. This contribution is realized in chapters 5, 6, 7, and 8.

1.6 Organization of the Thesis

The potential of the neural-based EDAs in solving MOPs served as a main motivation for the research work presented in this thesis. In order to achieve the aforementioned objectives, an MOEDA that uses an RBM as its modelling paradigm has been devised. The characteristics of the proposed algorithm in the perspective of evolutionary optimization and its implementation to handle MOPs with vastly different difficulties and problem nature are then presented.

The organization of the remaining chapters of this thesis is as follows. Chapter 2 presents an overview of the state-of-the-art MOEAs. The literature review of the multi-objective optimization

via EDAs is presented. Some other heuristic algorithms, which are widely considered in this thesis, are also introduced. This chapter also describes the performance metrics that are used to provide quantitative measurements of the generated results. The description of the test problems used for performance assessments follows.

Chapter 3 presents an RBM-based EDA (REDA) in the domination-based framework of multi-objective optimization. Clustering is incorporated to REDA. The performance of the algorithm is tested in test problems with a scalable number of objective functions and decision variables. The comparison and investigation results are presented in detail. Chapter 4 describes the behaviours of REDA in an evolutionary process. A sampling mechanism for REDA based on the energy functions of the RBM is highlighted.

Chapter 5 describes the implementation of REDA in noisy MOPs. A likelihood correction scheme is proposed in order to tune the modelled probabilistic distribution that has been distorted by the noisy objective functions. REDA is hybridized with a PSO algorithm in order to enhance its search ability. The experimental results, scalability issues, other possible hybridizations, and computational times are presented next.

Chapter 6 studies the optimization performance of REDA in solving multi-objective travelling salesman problem (MOTSP). Unlike the previous implementations where the test problems are in the real-number representation, the permutation-based representation is studied here. Chapter 7 extends the study of REDA in solving multi-objective multiple travelling salesman problem (MmTSP). Instead of using the domination-based framework of multi-objective optimization as in previous few chapters, this chapter designs REDA on the decomposition-based framework of multi-objective optimization.

Chapter 8 describes the ensemble algorithms between REDA, GA, and DE. The ensemble algorithms are developed in the domination-based and decomposition-based frameworks of multi-objective optimization. The effectiveness of the ensemble algorithms in finding a set of tradeoff Pareto optimal solutions are tested under various MOPs with different characteristics.

Finally, Chapter 9 presents the conclusions of this thesis and discusses future work.

Chapter 2

Literature Review

MOEAs have received a great deal of attention from the research community. Over the past few decades, many studies related to the algorithmic issues of MOEAs have been carried out. In this chapter, the MOEAs in different frameworks of multi-objective optimization are discussed. A review of the multi-objective estimation of distribution algorithms (MOEDAs) is also presented. Four state-of-the-art MOEAs that are seriously considered in this thesis are highlighted. The performance metrics that are used to provide a quantitative measurement of the obtained Pareto front are described. Finally, the test problems that are used to test the efficiency of MOEAs are presented.

2.1 Multi-objective Evolutionary Algorithms

In this section, several remarkable and state-of-the-art algorithms in different frameworks of multi-objective optimization are described.

2.1.1 Preference-based Framework

The preference-based framework of multi-objective optimization is the classical methods for handling MOPs. The basic idea of this framework is to aggregate the multiple conflicting objectives of MOPs into a single-objective optimization problem or to use preference knowledge of

the problems so that the optimizers can focus on optimizing certain objectives.

The most fundamental approach under this framework is to aggregate the multiple conflicting objectives into a single-objective through a weighted sum method [1]. This method combines all the conflicting objectives by multiplying each objective with a predefined weight value. Weighted metric or weighted Tchebycheff method is another approach that combines the multiple objectives into a single-objective. In this approach, the aim is to minimize the weighted distance metrics, where the distance metric measures the distance of a solution to the ideal solution. In [22], Haimes *et al.* proposed a method to only optimize one of the objectives and keep the other objectives within user-predefined values. In this method, different optimal solutions can be generated with different user-predefined values.

The main drawback of this framework is that it fails to achieve the common goal of multi-objective optimization, which is to obtain a set of tradeoff and diverse solutions, in a single simulation run. Furthermore, it is necessary to give preference knowledge of MOPs, such as suitable weight values or user-predefined values, to optimizers. The research of this framework mainly focuses on studying how to effectively employ the preference information in performing optimization. Detailed description of this framework can be referred to [1, 23–26].

2.1.2 Domination-based Framework

The algorithms in this framework optimize all conflicting objectives of MOPs simultaneously by assigning a fitness to each solution. This idea was first suggested by Goldberg [27]. However, he did not carry out simulation to prove the suitability of his idea in handling MOPs. The first remarkable MOEAs in this framework, the multi-objective genetic algorithm (MOGA), was proposed by Fonseca and Fleming [28]. In MOGA, two important steps have been designed to determine the fitness of a solution. First, the fitness of a solution is calculated according to the number of other solutions that dominates it. This fitness is used to determine the rank of the solutions. Second, the fitness of the solutions in the same rank is shared by a fitness sharing

mechanism. Using these two steps, MOGA is able to maintain a set of non-dominated solutions in a single simulation run.

Srinivas and Deb [29] employed a similar framework as MOGA and introduced a new ranking and sharing mechanism. The proposed algorithm is named as non-dominated sorting genetic algorithm (NSGA). Instead of counting the number of individuals that dominates each solution, Srinivas and Deb proposed a ranking mechanism that ranks the solutions according to the level of domination. The first level comprises of all non-dominated solutions. Then, the solutions marked as first level are ignored. The second set of non-dominated solutions in the population are identified and marked as the second level. This ranking continues until no more solutions are stored in the population. The diversity is maintained through a fitness sharing method. A stochastic remainder proportional selection scheme is used to select the solutions by giving a higher change to select solutions in a lower front.

The MOGA and NSGA suffer several limitations. First, they are non-elitist approaches. Second, it is necessary to specify a sharing parameter. Third, the NSGA has a higher computational complexity. In order to develop a more efficient algorithm for multi-objective optimization, some researchers incorporated an elitism mechanism, which is an external archive of solutions, into optimization algorithms.

In [30], Zitzler and Thiele proposed an elitist MOEA called strength Pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1999). An external archive is created to maintain a set of non-dominated solutions (external population) found during evolutionary processes. In every generation, a new non-dominated elite found in current population will be archived while the dominated solutions in the external population will be discarded. Once the external population reaches a maximum number of allowed solutions, the crowded solutions, determined by a clustering algorithm, will be discarded. As for fitness assignment, a fitness value will be assigned to both external and current populations. The fitness of a solution in the external population is proportional to the number of solutions in the current population that is being dominated by

the solution in the external population. On the other hand, the fitness of a solution in the current population is proportional to the sum of the fitness of solutions in the external population that dominates the solution in the current population. In [31], Zitzler *et al.* proposed SPEA2, which is an improved version of SPEA. In SPEA2, an improved fitness assignment, archiving, and diversity preservation mechanism were proposed.

In [32], NSGA-II, which is an improved version of NSGA, is proposed. NSGA-II also preserves a set of archived solutions from the beginning of evolution. Instead of storing the non-dominated solutions only as SPEA, NSGA-II store all the parent (N solutions) and children (N solutions) solutions. Subsequently, a non-dominated sorting is applied to the entire archive solutions ($2N$ solutions). The solutions are classified according to the rank of domination. The best non-dominated solutions are marked as first rank. The second best non-dominated solutions are marked as second rank, and so on. The best N solutions with lower ranks are selected as parent solutions which will undergo evolution in the next generation. Since only N number of solutions will be selected as parent solutions, some of the solutions in a particular rank will not be fitted to a parent population. In this case, a crowding distance measurement is used to determine the less crowded solutions to become parent solutions. Compared to NSGA, NSGA-II has lower computational complexity, uses an elitist approach, and does not need to specify a sharing parameter. Currently, NSGA-II is one of the most famous MOEAs that has been implemented to solve many real-world problems and serve as a baseline algorithm for MOEAs. However, the optimization performance of NSGA-II is poor in problems with more than three objectives. This is because both its ranking and crowding mechanisms are inefficient in differentiating the superiority of the solutions in many-objective problems.

The dominance-based framework of multi-objective optimization has become one of the main research areas over the past decade. The ability to obtain a set of tradeoff solutions in a single simulation run determines the appropriateness of this approach for multi-objective optimization. In addition, the diversity of the solutions could be preserved by considering the distribution

of the solutions in a maintained population. However, a major drawback of this approach is that the selective pressure is weakened with the increase in the number of objective functions. Therefore, this approach is only suitable for tackling problems with two or three objective functions. Furthermore, it is necessary to determine the setting of a sharing parameter in some diversity preservation schemes.

Since then, many variations of MOEAs in the domination-based framework have been proposed. Most of them follow a near-similar process flow as the aforementioned elitist MOEAs. In order to further enhance the optimization ability of MOEAs, several aspects of the domination-based algorithms have become main research concerns. Those aspects are indicator-based MOEAs [33,34], hybrid MOEAs [35,36], memetic MOEAs [37,38], coevolution-based MOEAs [39,40], adaptive MOEAs [41, 42], and parallel MOEAs [43, 44]. Furthermore, the implementation of the domination-based MOEAs have also vastly used to tackle various MOPs, including noisy MOPs [45, 46], dynamic MOPs [47, 48], constrained MOPs [49, 50], scalable MOPs [51, 52], many-objectives MOPs [53, 54], and multi-modal MOPs [55, 56]. In addition to employing GA as the search algorithm, many other EAs have been implemented in the domination-based framework of multi-objective optimization. They are differential evolution (DE) [57, 58], particle swarm optimization (PSO) [59, 60], EDAs [61, 62], ant colony optimization (ACO) [63, 64], artificial immune system (AIS) [65,66], genetic programming (GP) [67,68], and evolution strategy (ES) [69, 70]. A more detailed review of the domination-based MOEAs can be referred to [12, 18, 26].

2.1.3 Decomposition-based Framework

The decomposition-based framework decomposes an MOP into several subproblems; subsequently the algorithm optimizes all the subproblems concurrently. The Pareto dominance is partially applied or fully eliminated in this framework.

In [71], a multi-objective genetic local search algorithm (MOGLS) was proposed. In

MOGLS, the multiple objectives of an MOP are aggregated using a weighted sum approach. During the selection process, a pair of parent solutions is selected to perform genetic operations (crossover and mutation) in order to generate an offspring. The offspring is then optimized by using local search where the fitness function of the offspring is an aggregated weighted-sum function. In performing each local search, a new vector of weight information is randomly generated. In MOGLS, an external archive, which stores the non-dominated solutions, is also implemented. After a local search, if an offspring is non-dominated by any parent or archived solutions, the offspring is added to the archive. The solutions in the archive that are dominated by the newly added solutions are discarded. In this algorithm, Pareto dominance is partially applied.

In [72], another version of MOGLS was proposed. In this version, the aim was to simultaneously optimize an MOP instead of finding a non-dominated solution in each iteration as done in [71]. The multiple objective functions of an MOP are aggregated using a weighted Tchebycheff scalarizing function. In each generation, a randomly generated weight vector is implemented. Then, a set of solutions with the best scalarizing function are selected to form a temporary population. A pair of parent solutions is selected from the temporary population and genetic operations are performed to generate an offspring. Subsequently, the generated offspring is improved locally. The offspring is then updated to the main population if it has a better scalarizing fitness than the worst solution in the temporary population.

In [73], a two-phase local search for bi-objective optimization was proposed. This algorithm attempts to optimize an MOP by optimizing a set of modified objective functions. The modified objective functions are constructed using weighted linear utility aggregation function. The idea is inspired from the use of local search to optimize several scalar optimizing functions. After a scalar function is optimized, the next optimization is performed to the second scalar function. The process continues until all scalar functions are optimized. During the optimization process, the optimal solution obtained in the previous scalar function is used as the starting point for solving the next scalar function. This algorithm showed promising performance in solving bi-objective

TSP. However, the diversity issue has not been carefully studied and investigated.

In [74], Zhang and Li proposed an MOEA based on decomposition (MOEA/D). In MOEA/D, an MOP is decomposed into several subproblems in which each subproblem is constructed by using a weighted sum or Tchebycheff approach. Thus, each subproblem is a scalar objective optimization problem. For the aggregation issue, a set of uniformly weight vectors is generated. This weight vector serve as a criterion, in terms of Euclidean distance, to determine the neighbourhood relations between the subproblems. The subproblems are then optimized by only considering the neighbourhood solutions. This is done by performing genetic operations between a subproblem with its neighbourhood solutions. The best solution found, in terms of the aggregated fitness value, is then updated to that subproblem and its neighbourhood solutions. In MOEA/D, there is no particular pool for archiving and elitism. Even though there is no obvious elitism being applied, it is implicitly carried out when the nearest neighbours (parents) are updated by comparing their fitness values with those of the offspring (newly generated solution for a subproblem). Under this scheme, selective pressure in many-objective problems as faced by the domination-based MOEAs is solved since the fitness of a solution merely depends on the aggregated objective value. Besides, there is no need to specify a diversity preservation scheme since the diversity is preserved in the predefined weight vectors. The main advantage of MOEA/D is that a common algorithm for single-objective optimization can be directly applied to optimize the constructed subproblems while its framework can preserve a set of diverse solutions.

An improved version of MOEA/D was proposed in [75]. First, instead of using GA as the search algorithm, DE was integrated into MOEA/D to form a new algorithm called MOEA/D-DE. With DE operator, MOEA/D-DE is able to more effectively explore the search space especially when the Pareto set is complicated. This is because the DE operator, which creates an offspring by using three parent solutions, is able to generate a wide range of offspring. Thus, the exploration ability of the algorithm can be enhanced. Second, two neighbouring structures are suggested to balance the exploration and exploitation of the search. One of the structures

is that the maximum number of subproblems that are allowed to be updated by an offspring is equivalent to the neighbourhood size (T). This is identical to the neighbourhood structure in the original MOEA/D. This structure may reduce the diversity preservation if an offspring is greatly superior that its parent and its parent's neighbours. In order to reduce this effect, a second neighbouring structure is proposed. This structure bounds the number of subproblems to be updated by an offspring to T_B where T_B is much smaller than T.

In [76], a new MOEA/D with dynamic resource allocation (MOEA/D-DRA) was proposed and tested on CEC09 unconstrained MOPs test problems. In this version of MOEA/D, different computational efforts are assigned to different subproblems. This algorithm has been granted the best MOEA in the CEC09 competition for unconstrained multi-objective optimization. Since then, many researches have been carried out to improve MOEA/D or to implement MOEA/D in real-world problems. In [77], Palmers *et al.* suggested the use of more than one solution to represent each subproblem. In [44], the parallelism issue pertaining to MOEA/D was carried out. In [78], different aggregation functions were suggested at different search stages of MOEA/D. In real-world problems, MOEA/D have been implemented to study flowshop scheduling problems [79], antenna arrays [80], vehicle suspension optimization [81], and sensor network routing problems [82], just to name a few.

2.2 Multi-objective Estimation of Distribution Algorithms

EDAs were introduced as a new computing paradigm in the field of evolutionary computation [20]. In EDAs, offspring are produced by sampling the estimated probability distribution of the parent population. EDAs can be divided into different categories, according to the level of interaction between the decision variables that their probabilistic model performs [7]. Recently, several EDAs have been proposed in the context of multi-objective optimization.

Bosman and Thierens [83] presented a mixture distribution as a probabilistic model where each component in the mixture was a univariate factorization. This algorithm is known as a multi-

objective mixture-based iterated density estimation evolutionary algorithm (MIDEA). MIDEA has the advantage of preserving diversity due to its wide-spread exploration capability towards the Pareto optimal front. This algorithm can serve as a baseline algorithm for MOEDAs for its simplicity, speed, and effectiveness. In another study, Laumanns and Ocenasek [84] incorporated Bayesian optimization algorithm based on the binary decision tree as a model building technique - Bayesian multi-objective optimization algorithm (BMOA) - to approximate the set of Pareto optimal solutions. In [85], the authors proposed the multi-objective Parzen-based EDA (MOPED) to approximate the probability distribution of the solutions lying on the Pareto optimal front. Furthermore, a spreading technique which uses Parzen estimator in the phenotypic space to identify the crowding level of the solutions was also proposed. The algorithm showed promising results in several well-known test problems.

Li *et al.* [86] proposed a hybrid EDA (MOHEDA) by integrating a local search that is based on weighted sum method while its constraint handling is based on a random repair method for solving multi-objective 0/1 knapsack problems. A stochastic clustering method was also introduced to preserve the diversity of the solutions. The results showed that MOHEDA outperforms several state-of-the-art algorithms. Zhang *et al.* [21] proposed a regularity model-based multi-objective estimation of distribution algorithm (RM-MEDA) by considering the regularity in building the probabilistic model. A local principal component analysis was used to extract the regularity patterns of the candidate solutions from previous searches. This algorithm showed good performance in test instances with nonlinear variable linkage for a scalable number of variables. Okabe *et al.* [87] introduced Voronoi diagrams to construct the probability model in an algorithm called Voronoi-based EDA (VEDA). VEDA is able to adjust the reproduction process based on the problem structure and at the same time, it takes advantage of the problem structure by estimating the most appropriate probability distribution.

Pelikan *et al.* [88] described a scalable algorithm by combining non-dominated sorting genetic algorithm II (NSGA-II) [32], hierarchical Bayesian optimization algorithm (hBOA), and

clustering in the phenotypic space to solve decomposable problems. In the proposed algorithm, each cluster is allocated an almost equal number of solutions. The results show that the algorithm outperformed many standard MOEAs when the numbers of decision variables were scaled up. Another study was carried out by Zhong and Li [89] where a decision-tree-based multi-objective estimation of distribution algorithm (DT-MEDA) for continuous-valued optimization problems was developed. The conditional dependencies among the decision variables were extracted by a decision-tree-based probabilistic model. In [90], the authors implemented growing neural gas networks for modelling purposes. The algorithm showed promising results in high-dimensional test problems.

2.3 Related Algorithms

In this section, four state-of-the-art MOEAs, which are widely considered in this thesis, are presented.

2.3.1 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

NSGA-II [32] is one of the prominent algorithms using the domination-based framework of multi-objective optimization. The fitness assignment operators of NSGA-II will be implemented in most of the algorithms proposed in this thesis, thus, its operation is detailed in this section. The process flow of NSGA-II is presented in Figure 2.1.

Firstly, N solutions are generated to form an initial population $Pop(g = 0)$. All solutions in $Pop(g)$ are evaluated to obtain their objective values. Fitness is then assigned to all solutions in the population based on the Pareto ranking and crowding distance. Next, selection is applied to select a pair of parent solutions. By using the binary tournament selection, two chromosomes are randomly picked into tournament. A fitter solution in terms of lower rank or greater crowding distance is selected. Crossover is then performed to the selected pair of parent solutions to generate an offspring. The offspring will be exposed to random perturbation through the means

Begin
1. **Initialization:** At generation $g = 0$, randomly generate N solutions as the initial population, $Pop(g)$
2. **Evaluation:** Evaluate all solutions in $Pop(g)$

Do While ("Stopping Criterion is not satisfied")
3. **Fitness assignment:** Perform Pareto ranking and crowding distance to the population $Pop(g)$

For i=0 to N
4. **Selection:** Select parent solutions to perform genetic operations using binary tournament selection
5. **Crossover:** Perform crossover to the selected parent solutions to generate an offspring
6. **Mutation:** Perform mutation to the offspring

End for
7. **Evaluation:** Evaluate all offspring and store the offspring in an archive A
8. **Archiving:** Combine parent and offspring solutions $Pop(g) \cup A$
9. **Elitism:** Perform Pareto ranking and crowding distance to $Pop(g) \cup A$. Select the best N solutions to form new population $Pop(g+1)$. $g = g + 1$

End Do
10. **Output:** Output the final set of solutions $Pop(g)$

End

Figure 2.1: Pseudo-code of NSGA-II

of mutation. The procedures of selection, crossover, and mutation continue N times, so that N offspring are generated. Next, the offspring are evaluated and stored in a temporary archive A. In the archiving procedure, the parent and offspring solutions are combined to form a bigger archive with $2N$ size. In the elitism stage, Pareto ranking and crowding distance are performed to the solutions in the combined archive. Subsequently, the best N solutions are selected to form the new population $Pop(g+1)$. The best N solutions consist of solutions with lower Pareto ranks or greater crowding distances. This marks the completion of one generation. The same procedure is iterated over generations until a stopping criterion is met.

The illustration of the Pareto ranking and crowing distance is as follows. Solution X is fitter than solution Y if and only if all the objective values of the solution X are smaller than those of the solution Y (for the case of minimization problems). Solution X and solution Y are incomparable if and only if the objective values of the solution X are not all smaller than those of the solution Y. By applying this concept, the solutions are ranked according to their rank of domination. Solutions that are not being dominated by any other solutions are ranked as one (the lowest rank), while solutions that are only dominated by the solutions in rank one are ranked as

two, and so forth. In this case, the solutions in a lower rank are fitter than those in a higher rank. This ranking mechanism is illustrated in Figure 2.2. By applying natural selection pressure, the chances of selecting solutions in lower ranks are higher than solutions in higher ranks. However, if two solutions are located at the same rank, then the one that has a greater crowding distance is preferred. The crowding distance of a solution is calculated by summing over its two nearest neighbours. The procedure, illustrated in Figure 2.3, calculates the crowding distance of solution A by the summation of $L1$ to $L4$. The crowding distance for solutions at the boundary location (solutions B and C in Figure 2.3) is set to an infinite value, and this leads to the increase in the possibility of these solutions surviving a tournament selection indefinitely. This procedure aims to increase the ability of the algorithm to evolve a set of diverse solutions.

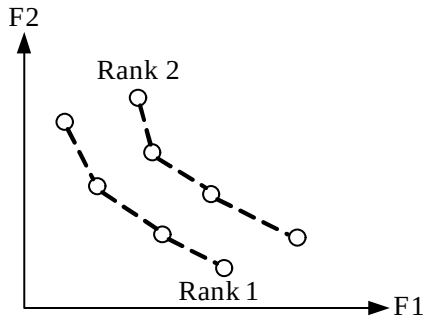


Figure 2.2: Pareto-based ranking. F1 is the first objective and F2 is the second objective.

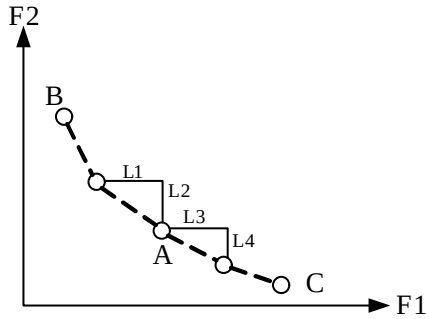


Figure 2.3: Crowding distance measurement. F1 is the first objective and F2 is the second objective.

In the binary-number representation, single point crossover is implemented. This crossover randomly cuts a chromosome into two pieces and then exchanges the pieces between two parent

solutions. The degree of crossover is controlled by a probability of crossover (p_c). For mutation, bit-flip mutation is employed. This is done by randomly flipping an allele with a probability of mutation (p_m).

In the real-number representation, simulated-binary crossover (SBX) is implemented. The SBX operator works as follows. Firstly, select two parents (x^{p1}, x^{p2}) from the population and generate a random value u between [0, 1]. Then the child solutions are created as follows:

$$x^{'} = 0.5[(x^{p1} + x^{p2}) - \beta^{'}|x^{p2} - x^{p1}|] \tag{2.1}$$

$$x^{'} = 0.5[(x^{p1} + x^{p2}) + \beta^{'}|x^{p2} - x^{p1}|] \tag{2.2}$$

$$\int_0^{\beta^{'}} \rho(\beta)d\beta = u$$

$$\rho(\beta) = \begin{cases} 0.5(\varphi+1)\beta^{\varphi} & \text{if } \beta \leq 1 \\ 0.5(\varphi+1)\frac{1}{\beta^{\varphi+2}} & \text{otherwise} \end{cases}$$

where β is the spread factor which follows a polynomial probability distribution $\varphi(\beta)$ and φ is the distribution index which gives the probability of creating child solutions that are distant from or near to the parent solutions. For the polynomial mutation operator, an offspring is created in the following way:

$$x^{'} = \begin{cases} x^{p1} + \sigma(x_L - x_U) & \text{if } u < p_m \\ x^{p1} & \text{otherwise} \end{cases} \tag{2.3}$$

$$\sigma = \begin{cases} (2u)^{1/(\varphi+1)} & \text{if } u < 0.5 \\ 1 - (2 - 2u)^{1/(\varphi+1)} & \text{otherwise} \end{cases} \tag{2.4}$$

where x_L and x_U are the lower and upper bounds for the decision variable x, respectively, φ is the distribution index, and u is a random value between [0, 1].

2.3.2 Multi-objective Univariate Marginal Distribution Algorithm (MOUMDA)

Univariate marginal distribution algorithm (UMDA) is one of the simplest and most famous EDAs proposed by Mühlenbein [91]. UMDA models the distribution of the selected population without considering any dependencies between the variables. UMDA closely follows the process flow of a typical EDA, thus can serve as a baseline algorithm for EDA. In this thesis, UMDA is constructed in the non-dominated sorting framework of NSGA-II and is called multi-objective UMDA (MOUMDA). The basic framework of the proposed algorithm in this thesis is similar to MOUMDA except that a different modelling mechanism is employed. The comparison results with this algorithm show the advantages of the proposed modelling mechanism. The pseudo-code of MOUMDA is presented in Figure 2.4.

Begin

1. **Initialization:** At generation $g = 0$, randomly generate N solutions as the initial population, $Pop(g)$
2. **Evaluation:** Evaluate all solutions in $Pop(g)$

Do While ("Stopping Criterion is not satisfied")

3. **Fitness assignment:** Perform Pareto ranking and crowding distance to the population $Pop(g)$
4. **Selection:** Select N parent solutions using binary tournament selection
5. **Modeling:** Estimate the probability distribution of the selected parent solutions $p_g(x)$
6. **Sampling:** Sample N offspring from $p_g(x)$ using simple sampling technique
7. **Evaluation:** Evaluate all offspring and store the offspring in an archive A
8. **Archiving:** Combine parent and offspring solutions $Pop(g) \cup A$
9. **Elitism:** Perform Pareto ranking and crowding distance to $Pop(g) \cup A$. Select the best N solutions to form new population $Pop(g+1)$. $g = g + 1$

End Do

10. **Output:** Output the final set of solutions $Pop(g)$

End

Figure 2.4: Pseudo-code of MOUMDA

The difference between NSGA-II and MOUMDA is in the Steps 4-6 of Figure 2.4. In the selection process, N parent solutions are picked using the binary tournament selection. Sub-

sequently, the marginal probability distribution of all possible alleles in each decision variables (x_i, $i \in \{1, ..., n\}$, n is the number of decision variables) of the selected set of parent solutions is estimated. In the binary-number representation, the marginal probability distribution of x_i is constructed as follows:

$$p_g(x_i) = \frac{\sum_{j=1}^{N} \delta_j(x_i) + 1}{N + r_i} \tag{2.5}$$

$$\delta_j(x_i) = \begin{cases} 1 & \text{if } x_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

where r_i is the number of different values that x_i may take. In the binary case, r_i=2. The delta function $\delta_j(x_i)$ is the value of variable x_i in the j^{th} individual. By using the constructed probabilistic model, N offspring are sampled by the simple sampling technique as follows:

$$x_i^{'} = \begin{cases} 1 & \text{if random } (0, 1) \leq p_g\,(x_i = 1) \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

2.3.3 Non-dominated Sorting Differential Evolution (NSDE)

Differential evolution (DE) is another EA that has received significant interest from the research community. In DE, three parent solutions are used to create an offspring. Due to this property, DE is claimed to be able to generate a wide set of offspring and has better exploration ability [75]. In [92], Iorio and Li proposed a non-dominated sorting differential evolution (NSDE), which is directly modified from NSGA-II by replacing the crossover and mutation operators of a GA with the operators of a DE. Thus, the process flow is exactly similar to Figure 2.1. The only difference between NSGA-II and NSDE is the implementation of different type of crossover operators.

The most fundamental crossover operator of a DE (DE/rand/1/bin), which was used in NSDE, is illustrated. In the selection process, three parent solutions (x^{p1}, x^{p2}, x^{p3}) are selected using the binary tournament selection. Then, an offspring is created as follows:

$$x^{'} = \begin{cases} x^{p1} + F(x^{p2} - x^{p3}) & \text{if } u < p_c \\ x^{p1} & \text{otherwise} \end{cases} \tag{2.7}$$

where F is a scalar number, p_c is the probability of performing the DE operator, and u is a random value between [0, 1]. After that, the polynomial mutation is applied to the generated offspring. The selection, crossover, and mutation procedures are repeated N times. Subsequently, the evaluation, archiving, and elitism, similar to NSGA-II, are iteratively carried out until the stopping criterion is satisfied.

2.3.4 MOEA with Decomposition (MOEA/D)

The MOEA/D proposed in [74] is extensively used in this thesis either as an algorithm for the comparison or a reference model for the decomposition-based MOEA. The pseudo-code of MOEA/D is presented in Figure 2.5.

Initially, a set of uniformly distributed weight vectors, $\boldsymbol{\lambda}^1$,...,$\boldsymbol{\lambda}^N$, is generated. N is the predefined size of subproblems. Then, the Euclidean distance between all weight vectors is calculated. For each weight vector i, the Q neighbouring solutions ($B(i) = \{i_1, ..., i_Q\}, i \in [1, N]$) that are closer, in terms of Euclidean distance, to weight vector i, are determined. Next, N solutions are randomly generated to be the initial population $Pop(g = 0)$. All solutions in $Pop(g)$ are then evaluated to obtain their objective values FV. The best objective values of the population are used as a starting reference point ($\mathbf{z}^*$) of the Tchebycheff approach.

The Tchebycheff approach is the aggregation technique used in MOEA/D to decompose an MOP into N scalar optimization subproblems. For subproblem j, the aggregated function is presented as follows:

Begin

1. Initialization

a) Generate a set of uniformly distributed weight vectors ($\boldsymbol{\lambda}^i, \ldots, \boldsymbol{\lambda}^N$)

b) Calculate the Euclidean distance among the weight vectors. Determine the Q neighboring solutions ($B(i) = \{i_1, \ldots, i_Q\}, i \in [1, N]$) for each weight vectors according to the shortest Euclidean distance.

c) At generation $g = 0$, randomly generate N solutions to be the initial population, $Pop(g = 0)$. Evaluate the solution and set $FV^i = \boldsymbol{f}(\boldsymbol{x}^i), i \in 1, \ldots, N$

d) Initialize reference point of the Tchebycheff approach ($\boldsymbol{z}^*$) by setting the value of $\boldsymbol{z}^*$ to be the lowest objective values of the solutions

Do while ("Stopping Criterion is not satisfied")

For $\boldsymbol{i} = \mathbf{1}$: $\boldsymbol{N}$

2. Selection: Randomly select two solutions from $B(i)$

3. Crossover: Perform crossover to the parent solutions to generate an offspring

4. Mutation: Perform mutation to the generated offspring

5. Evaluation: Evaluate the generated offspring ($\boldsymbol{y}$) to obtain the corresponding objective values, $\boldsymbol{f}(\boldsymbol{y})$

6. Update of $\boldsymbol{z}^*$: For $j = 1, \ldots, m$, if $z_j^* > f_j(\boldsymbol{y})$, then set $z_j^* = f_j(\boldsymbol{y})$

7. Fitness assignment: Assign fitness (g^{te}) to each solution using Tchebycheff approach

8. Update Solution: For $j \in B(i)$, if $g^{te}(\boldsymbol{y}|\boldsymbol{\lambda}^j, \boldsymbol{z}^*) \leq g^{te}(\boldsymbol{x}^j|\boldsymbol{\lambda}^j, \boldsymbol{z}^*)$, then set $\boldsymbol{x}^j = \boldsymbol{y}$ and $FV^j = \boldsymbol{f}(\boldsymbol{y})$

End For

End Do

End

Figure 2.5: Pseudo-code of MOEA/D

$$g^{te}(\mathbf{x}|\boldsymbol{\lambda}^j, \mathbf{z}^*) = \max_{1 \leq i \leq m} \{\lambda_i^j | f_i(\mathbf{x}) - z_i^* |\} \tag{2.8}$$

where m is the number of objective functions and f_i is the objective value for the i^{th} objective function. In the evolutionary process, MOEA/D aims to minimize all N subproblems concurrently in a single simulation run.

Next, the evolution loop starts. For each subproblem i, two neighbouring solutions are randomly selected from $B(i)$. Then, crossover to the selected parent solutions is performed to generate an offspring. The SBX is used as the crossover operator. The offspring are then exposed to undergo polynomial mutation. The process continues with the evaluation of the generated offspring $\mathbf{y}$ to obtain the corresponding objective value $\mathbf{f}(\mathbf{y})$. Then, $\mathbf{z}^*$ is updated if $z_j^* > f_j(\mathbf{y})$. For all subproblems and offspring, the fitness (g^{te}) of the aggregated scalar function is calculated. For each neighbouring solution j for subproblem i, $j \in B(i)$, if the fitness of offspring $\mathbf{y}$ is better

than the j^{th} neighbouring solution, then the j^{th} neighbouring solution is replaced by the offspring $\mathbf{y}$. The same procedures of selection, reproduction, update of $\mathbf{z}^*$, fitness assignment, and update of solution are repeated until a stopping criterion is satisfied.

2.4 Performance Metrics

Instead of finding a single optimum solution as done in single-objective optimization, multi-objective optimization aims to find a set of tradeoff solutions that satisfy four common goals of multi-objective optimization, including proximity, diversity, uniformity, and number of non-dominated solutions. In this case, performance metrics are important tools that provide a scalar quantitative measurement of the generated set of solutions. The issue pertaining to the performance metrics have been discussed by many researchers [12, 72, 93–96]. Four performance metrics or indicators that are commonly used to show the numerical comparison for different goals of multi-objective optimization between different algorithms are applied in this thesis. The performance metrics are:

1. **Generational Distance (GD):** This indicator is a representative metric which provides a quantitative measurement for the proximity goal of multi-objective optimization.

2. **Maximum Spread (MS):** This indicator is a representative metric which provides a quantitative measurement for the diversity goal of multi-objective optimization.

3. **Inverted Generational Distance (IGD):** This indicator is a representative metric which provides a quantitative measurement for the proximity and diversity goal of multi-objective optimization.

4. **Non-dominated Ratio (NR):** This indicator is a representative metric which provides a quantitative measurement of the number of non-dominated solutions for the goal of multi-objective optimization.

The mathematical description of the performance metrics is given in Appendix A.

2.5 Test Problems

Benchmark test problems, where the Pareto optimal are known, are an important element to test the ability, efficiency, and possible pitfalls of an MOEA. In order to effectively test an MOEA, the test problems should consist of different characteristics and difficulty levels. In the literature, many test problems for MOPs have been designed [97–100]. Here, 31 state-of-the-art test problems, which are used in this thesis, are presented. The test problems are five ZDT test problems [101], seven DTLZ test problems [102], 10 UF test problems [103], and nine WFG test problems [104]. The characteristics, difficulty levels, and mathematical formulation of the test problems are presented in Appendix B.

2.6 Summary

In this chapter, a literature review of MOEAs has been presented. The review classified MOEAs into three main categories, according to the ways the MOEAs handle the multiple conflicting objectives of MOPs. They are MOEAs with the preference-based framework, domination-based framework, and decomposition-based framework. Several state-of-the-art algorithms in different frameworks of multi-objective optimization were discussed. Furthermore, the EDAs for multi-objective optimization were outlined. This chapter also described four state-of-the-art MOEAs, which are widely considered in this thesis, in detail. The performance metrics and benchmark test problems that are used in this thesis were also presented.

Chapter 3

An MOEDA based on Restricted Boltzmann Machine

The exploitation of linkage information between the decision variables of an MOP in guiding the search towards optimality is one of the main characteristics of estimation of distribution algorithms (EDAs). In this chapter, the restricted Boltzmann machine (RBM) is modelled as a novel EDA in the context of multi-objective optimization. RBM is an energy-based stochastic neural network. The probabilities of the joint configuration over the visible and hidden units in the network are trained using contrastive divergence until the distribution over the global state reaches a certain level of thermal equilibrium. Subsequently, the probabilistic model is constructed using the energy function of the network. In addition, clustering in phenotypic space is incorporated into the proposed algorithm. The effects on clustering and the stability of the trained network on optimization performance are rigorously examined. Experimental studies are conducted to analyze the performance of the proposed algorithm in scalable problems with large number of objective functions and decision variables.

3.1 Introduction

EDAs have been recognized for being able to perform better on some problems where genetic algorithms (GAs) fail to give satisfactory performance [105–107]. However, the performance of EDAs is inherently dependent upon the probabilistic modelling and sampling technique. Univariate modelling is simple and easy to implement, but does not utilize linkage information to guide the search. This may hinder the algorithm when solving complex problems. Bivariate or multivariate modelling improves the ability of algorithms by using linkage information to explore the search space; but with increased complexity and computational time [7, 108]. In the recent past, many variations of EDAs have been developed in the context of multi-objective optimization - multi-objective estimation of distribution algorithms (MOEDAs) [7, 8]. Most of the proposed modelling approaches in the literature used statistical inference to extract the statistical information of the selected population. However, modelling from statistical methods is complicated. It will also be hard to derive accurate information in this manner from complex problems if the input data is small.

Many real-world optimization problems are challenged by the unknown characteristics of the problems. The problems may be non-linear, constraint, has complex relationships within the variables, large number of variables, and even consists of several conflicting objectives [109–113]. High-dimensional problems with many decision variables and conflicting objective functions to be optimized simultaneously are hard problems which may challenge the algorithm in finding the global optimal solutions. In problems with many decision variables, the complexity of the problems would increase with an increase in the number of decision variables. This is due to the enlargement of the search space and an increase in the number of possible moves towards optimality.

Selection pressure in selecting fitter individuals is reduced when problems consist of many conflicting objective functions (more than three). This is due to the high rate of non-dominance among individuals during the evolutionary process. This may hinder the search towards optimal-

ity or result in the population getting trapped in a local optimal. One of the ideas to overcome these issues is to exploit extra information (e.g. linkage dependencies) from within the selected population. This information is hypothesized to provide valuable guidance in driving the search process. EDAs, which are characterized by their ability to capture linkage information between solutions, may be one of the promising algorithms in dealing with scalable problems. In the literature, most of the proposed EDAs for scalable problems only examine scalability in either objective function or decision variable, but not both.

This chapter attempts to address the issues discussed above by modelling the restricted Boltzmann machine (RBM) as an EDA to solve scalable multi-objective optimization problems. RBM [114–116] is a kind of neural network that learns the probability distribution in terms of energy equilibrium. A two-layer network with undirected graph in RBM is used to model the energy function of the equilibrium state. This information is captured through pair-wise interactions between visible and hidden units. Hidden units are functional for training capacity, in capturing the linkage dependencies between the dimensions of the input data. Contrastive divergence (CD) is employed to train the network until the distribution over the global state reaches a certain level of thermal equilibrium. The probabilistic model is subsequently constructed from the trained network. After which, new offspring are generated from the constructed probabilistic model. Empirical studies were carried out to investigate the performance and potential of RBM in building the representative probabilistic model in solving scalable MOPs. Clustering in objective space using k-mean clustering is incorporated into the algorithm. The effects on the stability of the trained network and clustering in optimization are also rigorously examined. A good model for constructing the probability distribution is one that is easy to build and sample, and returns solutions with good fidelity [21]. Restricted Boltzmann machine-based multi-objective evolutionary algorithm (REDA) satisfied these criteria since the statistical information is gained through a learning process and empirical results show that the algorithm is effective in solving scalable problems with large number of objectives and decision variables.

The remainder of this chapter is organized as follows. Section 3.2 presents the existing work that implement MOEDAs to study the scalability issue of multi-objective optimization. Section 3.3 introduces the network architecture of RBM and its training algorithm. Section 3.4 presents the algorithmic framework of REDA, its modelling, and sampling method. Test instances, other algorithms in the comparison, and implementation are outlined in Section 3.5. Section 3.6 presents the experimental results and a summary is provided in Section 3.7.

3.2 Existing studies

Over the last decade, several estimation of distribution algorithms have been developed in the context of multi-objective optimization (MOEDAs). The main difference among the algorithms is the employment of different probabilistic modelling mechanisms. These algorithms have been used to deal with different types of problems. Among them, scalable problems have gained remarkable attention. The overview of the existing MOEDAs have been presented in Section 2.2. This section will discuss the MOEDAs that were designed to deal with scalable problems.

Two types of scalable problems have been studied in the literature: scalability in decision variables and scalability in objective functions. Scalability in decision variables increases the difficulty of the problem since the number of possible combinations towards optimality is increased with an increase in the number of variables. Besides, less selection pressure is devoted to each non-dominated individual in problems with many objective functions (more than three). This is caused by the fact that most of the individuals in the population are non-dominated solutions even in the early stages of evolution, making it more difficult for the algorithm to decide which individual is fitter. Both of these problems remain less explored, especially when dealing with high dimensionality in the objective space. In [1], the authors have shown that optimization algorithms require an exponential increase in the number of individuals with increasing number of objective functions. In order to reduce the number of individuals required for evolution at any time, EDAs capture the linkage information and use it to guide the search. As such, EDAs are

hypothesized to perform better than other EAs with stochastic genetic operators.

In [88], Pelikan *et al.* proposed an EDA based on hierarchical Bayesian optimization algorithm (hBOA) to solve multi-objective decomposable problems. The k-mean clustering and other operators from non-dominated sorting genetic algorithm-II (NSGA-II) were adapted in hBOA. Deceptive problems with scalable decision variables were used to examine the effectiveness of the proposed algorithm. Experimental results indicated that clustering and linkage learning are the main criteria which contribute to the success of the algorithm in solving decomposable multi-objective problems. In another study, Sastry *et al.* [117] presented an MOEDA based on extended compact genetic algorithm (ECGA). The paper analyzed the characteristics of the algorithm on a class of bounding adversarial problems with scalable decision variables. m-k deceptive trap problems [88] were used to investigate the performance of the proposed algorithm.

In [118], Soh and Kirley proposed a parameter-less MOEA which combines the ECGA with external ϵ-Pareto archive, clustering, and competent mutation to deal with scalable problems. Two types of scalable problems were addressed, including deceptive problems with scalable decision variables and DTLZ problems with scalable objective functions. The proposed algorithm showed promising results due to the combination of linkage learning, clustering, and local search. In [21], Zhang *et al.* have tested the RM-MEDA in high-dimensional test problems. The results indicated that the RM-MEDA scaled well and took less fitness evaluations to reach a predefined IGD value. Marti *et al.* [90] developed a new MOEDA based on growing neural gas (GNG) network and it was named as multi-objective neural EDA (MONEDA). GNG network is a self-organizing neural network based on neural gas model. This model creates an ordered cluster of input dataset; a new cluster will then be inserted based on the topology and cumulative errors. WFG problems [104] were chosen to evaluate the search capability of the MONEDA.

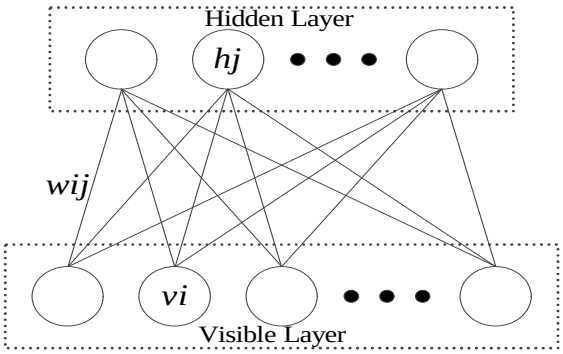


Figure 3.1: Architecture of an RBM

3.3 Restricted Boltzmann Machine (RBM)

RBM [114, 119, 120] is an energy-based stochastic binary neural network. It has been increasingly gaining research interest from the neural network community as a feature extraction method [120, 121].

3.3.1 Architecture of RBM

The architecture of the network is illustrated in Figure 3.1. The network is made up of stochastic binary neurons in a two-layered undirected graphical model comprising of a visible layer and a hidden layer. The visible layer, denoted as v_i, is an input layer of the network. The hidden layer, denoted as h_j, is a latent layer that determines the capability of the network in modelling the probability distribution of the input stimuli. The network does not have the output layer. Instead, the output information is represented by the energy values of the network. w_{ij} is the weight of the connection between the visible unit i and the hidden unit j. b_i is the bias of the visible unit i and d_j is the bias of the hidden unit j. Both of the layers are fully connected to one another and the weights are symmetric. In this way, the information can flow from one layer to another, increasing the learning capability of the network. Furthermore, there is no interconnection between the neurons within the same layer. Thus, the hidden units are conditionally independent. Besides, the visible units can be updated in parallel given the hidden states. This behaviour improves the training speed of the network. The weights and biases of an RBM define the energy function of

the network. The energy function is presented as follows:

$$E(v,h) = -\sum_i \sum_j v_i h_j w_{ij} - \sum_i v_i b_i - \sum_j h_j d_j \tag{3.1}$$

As in a Hopfield net, the energy is determined by the biases and weights. Through the energy function, the probability distribution of any global states can be derived as follows:

$$p(v,h) = \frac{e^{-E(v,h)}}{Z} \tag{3.2}$$

where Z is the normalizing constant

$$Z = \sum_{x,y} e^{-E(x,y)} \tag{3.3}$$

The probability distribution defined in equation (3.2) is measured over both visible and hidden units depending on the energy of the joint configuration compared with the normalizing constant. The normalizing constant is the energy of all the joint configurations. The probabilities of the visible units (data point) are the sum of the probabilities of all the joint configurations that contain it, as given in the following equation:

$$p(v) = \sum_h p(v,h) = \frac{\sum_h e^{-E(v,h)}}{Z} \tag{3.4}$$

3.3.2 Training

The network will be trained until it reaches a certain level of thermal equilibrium. At thermal equilibrium, the probability distribution of the global states converges. The problem lies in the training of the network to ensure that the energy level reaches or fluctuates around global min-

imum. Thus, the training method for RBM is crucial. Contrastive divergence (CD) is probably the most popular and efficient training method [115, 119].

In the CD training, two phases (positive phase and negative phase) are carried out. In the positive phase, the input stimuli or input data are rendered into the visible units of the network. Subsequently, the hidden states, given the visible states, are constructed by performing the Gibbs sampling according to the following equation:

$$p(h_j|v) = \varphi(\sum_i w_{ij}v_i - d_j) \tag{3.5}$$

where $\varphi(x) = \frac{1}{1+e^{-x}}$ is the logistic function. In the negative phase, given the hidden states, the visible states are reconstructed using the same logistic function as given in the following equation:

$$p(v_i|h) = \varphi(\sum_j w_{ij}h_j - b_i) \tag{3.6}$$

The process of these two phases is repeated S times. Next, the weights and biases of the network are updated as follows:

$$w_{ij}^{'} = w_{ij} + \epsilon(< v_i h_j >_0 - < v_i h_j >_S) \tag{3.7}$$

$$b_i^{'} = b_i + \epsilon(< v_i >_0 - < v_i >_S) \tag{3.8}$$

$$d_j^{'} = d_j + \epsilon(< h_j >_0 - < h_j >_S) \tag{3.9}$$

where ϵ is the learning rate, $<>_0$ is the original states of the neurons, and $<>_S$ is the expected

states of the neurons after S-step reconstruction. S is typically set to one during the initial stages of training and then increased gradually as the learning converges. Large values of S may approximate the maximum likelihood learning arbitrarily well. However, for simplicity, S could be set to one and then performs the training iteratively [122]. The overall CD training is repeated until a stopping criterion is met. The process of CD training is further demonstrated in Figure 3.2.

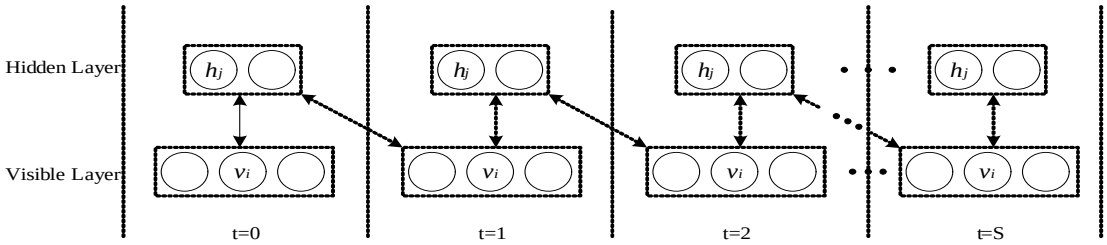


Figure 3.2: Contrastive divergence (CD) training

3.4 Restricted Boltzmann Machine-based MOEDA

3.4.1 Basic Idea

EDAs differ from standard genetic algorithms (GAs) due to the absence of crossover and mutation operators. These operators are replaced by a probabilistic model of the selected population and a sampling mechanism. RBM, an energy-based stochastic neural network, is suitable for building the probabilistic model for EDAs since the probabilities of the joint configuration over the visible and hidden layers are proportional to the energy of the joint configuration as given in the following equation:

$$p(v, h) \propto e^{-E(v,h)} \tag{3.10}$$

Furthermore, the dependencies between the decision variables are learnt through the training

process and the information is stored as connection weights and biases. This feature is expected to steer the search towards the Pareto optimal front since linkage information is considered in the modelling of the probability distribution.

3.4.2 Probabilistic Modelling

In the implementation stage, the alleles of the decision variable in the cost function are the input data to be rendered to the visible layer of an RBM. Therefore, n variables in fitness functions means n visible units in RBM. The setting of the number of hidden units is to be determined by users. The complexity of the network will increase with the number of visible and hidden units. Since the probability distribution of the population needs to be constructed at every generation, it is essential for the model to be kept simple. Therefore, the number of hidden units is set to as small as possible as long as the probability model is representative.

After performing the CD training, a set of trained weights, biases, and hidden states are obtained. Subsequently, in binary representation, the joint probability distribution with n decision variables in generation g is formulated as follows:

$$P_g(v) = \prod_{i=1}^{n} p_g(v_i) \tag{3.11}$$

where $p_g(v_i)$ is the marginal probability of decision variable i (v_i) at generation g. The marginal probability of each decision variable is obtained through equation (3.4). Expanding the equation,

$$p_g(v_i = 1) = \frac{\sum_{l=1}^{N} \delta_l(v_i^+)}{\sum_{l=1}^{N} \delta_l(v_i^+) + \sum_{l=1}^{N} \delta_l(v_i^-)} \tag{3.12}$$

$$\delta_l(v_i^+) = \sum_{j=1}^{H} e^{-E(v_i^l=1,h_j)} \tag{3.13}$$

$$\delta_l(v_i^-) = \sum_{j=1}^{H} e^{-E(v_i^l=0,h_j)} \tag{3.14}$$

where $\delta_l(v_i^+)$ is the marginal cost of v_i when the cardinality of $v_i = 1$, $\delta_l(v^-)$ is the marginal cost of v_i when the cardinality of $v_i = 0$, N is the number of selected solutions or parent solutions, and H is the number of hidden units. Direct sampling from the above probabilistic model reaches a limit in progress when the probability reaches a maximum value of 1.0 or a minimum value of 0.0. Therefore, the lower and upper bounds are added to the probability distribution based on the average cost of cardinality. The modified version of the marginal probability is given as follows:

$$p_g(v_i = 1) = \frac{\sum_{l=1}^{N} \delta_l(v_i^+) + avg\left(\sum_{l=1}^{N} \delta_l(v_i)\right)}{\sum_{l=1}^{N} \delta_l(v_i^+) + \sum_{l=1}^{N} \delta_l(v_i^-) + r_i \times avg\left(\sum_{l=1}^{N} \delta_l(v_i)\right)} \tag{3.15}$$

where $avg\left(\sum_{l=1}^{N} \delta_l(v_i)\right) = \frac{\sum_{l=1}^{N} \delta(v_i)}{N}$ and r_i is the number of different values that v_i may take. In binary case, r_i is 2.

3.4.3 Sampling Mechanism

The child solutions are generated through the sampling of the constructed probabilistic model (a simple sampling technique) as follows:

$$v_i = \begin{cases} 1 & \text{if random } (0,1) \leq p_g\left(v_i = 1\right) \\ 0 & \text{otherwise} \end{cases} \tag{3.16}$$

where random (0,1) is a randomly generated value between [0, 1].

3.4.4 Algorithmic Framework

The proposed algorithm for multi-objective optimization called restricted Boltzmann machine-based MOEDA (REDA) can adapt any operators (ranking, selection, archiving, elitism) in stan-

dard MOEAs by replacing the crossover and mutation operators with RBM. The other operations including non-dominance ranking, crowding distance, binary tournament selection, and parent-child archiving, are implemented as in NSGA-II [32]. Moreover, k-mean clustering in phenotypic space is employed. Clustering is incorporated into the algorithm to built the probabilistic model from different regions in the search space. This has been proven to improve the optimization performance [83, 88].

Begin

1. **Initialization:** At generation $g = 0$, randomly generate N solutions as the initial population, $Pop(g)$
2. **Evaluation:** Evaluate all solutions in $Pop(g)$

Do While ("Stopping Criterion is not satisfied")

3. **Fitness assignment:** Perform Pareto ranking and crowding distance to the population $Pop(g)$
4. **Selection:** Select N parent solutions using binary tournament selection
5. **Training:** Train the RBM using CD training mechanism to obtain the weights and biases
6. **Clustering:** Group the selected parent solutions into k clusters using k-mean clustering
7. **Modeling:** Compute the probability distribution of the solutions in each cluster $p_g^1(\boldsymbol{x}), \ldots, p_g^k(\boldsymbol{x})$
8. **Sampling:** Sample N offspring from $p_g^1(\boldsymbol{x}), \ldots, p_g^k(\boldsymbol{x})$ using simple sampling technique
9. **Evaluation:** Evaluate all offsprings and store the offsprings in an archive A
10. **Archiving:** Combine parent and offspring $Pop(g) \cup A$
11. **Elitism:** Perform Pareto ranking and crowding distance to $Pop(g) \cup A$. Select the best N solutions to form new population $Pop(g+1)$. $g = g + 1$

End Do

12. **Output:** Output the final set of solutions $Pop(g)$

End

Figure 3.3: Pseudo-code of REDA

The pseudo-code of the overall algorithm is presented in Figure 3.3. The REDA works as follows. Firstly, N initial individuals $Pop(0)$ (at generation $g = 0$) are randomly generated. All the individuals in $Pop(g)$ are evaluated to obtain their corresponding fitness values. Based on their fitness values, Pareto ranking and crowding distance are performed. Next, a new population (N individuals) is chosen by performing the binary tournament selection on $Pop(g)$. An RBM is trained by using the CD training method until the stopping criterion is reached. k-mean clustering algorithm [46] is then applied to group the individuals into k clusters. The probabilistic model

in each cluster is subsequently built by computing the marginal distribution of each decision variable. From the probabilistic models, N new solutions are sampled to form new population A. Each probabilistic model will sample similar number of offspring. Elitism is carried out by combining the offspring (A) and parent ($Pop(g)$) to form a mating pool with $2N$ individuals. N new solutions are selected from $A \cup Pop(g)$ to form $Pop(g+1)$. One generation is completed here. The same procedure is carried out in subsequent generations until the stopping criterion or the maximum number of fitness evaluations is reached.

3.5 Problem Description and Implementation

Eight benchmark test problems (ZDT1, ZDT2, ZDT3, ZDT6, DTLZ1, DTLZ2, DTLZ3, and DTLZ7) are selected to test the effectiveness of the REDA in terms of converging to the true Pareto optimal front and maintaining a set of diverse solutions. ZDT (ZDT1, ZDT2, ZDT3, and ZDT6) test problems [9] have two objective functions and a scalable number of decision variables. They are characterized by convex, non-convex, discontinuous convex, and non-uniform convex Pareto optimal fronts, respectively. DTLZ (DTLZ1, DTLZ2, DTLZ3 and DTLZ7) [123] are scalable problems in terms of the number of objectives and variables. They are characterized by linear, spherical, many local optimal, and disconnected Pareto optimal fronts.

In this implementation, two performance indicators (inverted generational distance (IGD) and non-dominance ratio (NR)) have been chosen to show the numerical comparison between the REDA and three other state-of-the-art algorithms. IGD allows the comparison of the proximity and diversity performance indicators between the Pareto optimal Front (PF) and the simulated solutions. A lower IGD value indicates better performance. Meanwhile, NR measures the percentage of non-dominated solutions among the entire pool of solutions obtained by all the algorithms, and a higher value indicates that more non-dominated solutions are generated by the algorithm.

Three state-of-the-art algorithms (NSGA-II, MOUMDA and RM-MEDA) are chosen for

performance comparison with the REDA. NSGA-II [32] is a popular algorithm in evolutionary multi-objective optimization due to its ability to achieve promising solutions for most of the MOPs. This algorithm uses Pareto ranking and crowding distance as fitness assignment operators, binary tournament selection, uniform crossover, bit-flip mutation, and parent-offspring archiving. k-mean clustering is incorporated into the algorithm for fair comparison with the REDA. This algorithm is chosen because it generally gives good performance for most of the test problems and the REDA adapts most of its operators.

Multi-objective univariate marginal distribution algorithm (MOUMDA) is a simple EDA which is modified directly from NSGA-II by replacing the genetic operators with probabilistic modelling based on univariate marginal distribution [91, 124]. The basic architecture of MOUMDA is very similar to the MIDEA proposed by Bosman and Thierens [83]. One difference between MOUMDA and MIDEA is that clustering is based on k-mean clustering instead of leader algorithm. UMDA models the distribution of the selected population without considering any dependencies between the variables. The basic framework of REDA is similar to this algorithm except that linkage information is taken into consideration. The comparison results with this algorithm may show the advantages of using linkage information in dealing with scalable problems.

Lastly, regularity model-based multi-objective estimation of distribution algorithm (RM-MEDA) [21] is one of a new MOEDAs which models the regularity pattern of the problems by constructing the probability distribution of promising solutions. The local principle component analysis (PCA) is used to model the probability distribution of the promising individuals. The authors have proven the ability of this algorithm in dealing with scalable problems with nonlinear variable linkages. The codes provided by the authors were directly implemented for this chapter's experimental investigations.

A comparative study of REDA, MOUMDA, NSGA-II, and RM-MEDA is carried out to examine their performance in problems with a large number of decision variables and objective

functions. The indices of the algorithms as used in the box-plot are explained in Table 3.1. All algorithms were implemented in C++ and ran on an Intel Core 2 Duo, 3.0 GHz personal computer. The experimental settings are summarized in Table 3.2.

Table 3.1: Indices of the algorithms

Index	Algorithm
1	REDA
2	NSGA-II
3	MOUMDA
4	RM-MEDA

Table 3.2: Parameter settings

Parameter	Setting
Population size	100 for ZDT problems, 200 for DTLZ problems with 3 objectives, 500 for DTLZ problems with 5 objectives, and 1500 for DTLZ problems with 7 objectives
Number of hidden units for REDA	20 for ZDT problems and 5 for DTLZ problems
Number of training epochs for REDA	10 for all test instances
Learning rate for REDA	0.1 for all test instances
Number of clusters	1 for ZDT problems and DTLZ with 3 objectives, 7 for DTLZ problems with 5 objectives, and14 for DTLZ problems with 7 objectives
Stopping criterion	200 generations for all test instances
Mutation for NSGA-II	1/(Variable_size $\times$ Variable_bit)
Crossover for NSGA-II	0.8
Number of bits per variable	10 bits (REDA, MOUMDA and NSGA-II), real-coded representation for RM-MEDA
Number of independent runs	10
Number of decision variables	scalable for ZDT problems and $n = m + K - 1$ for DTLZ problems, where m is the number of objectives and $K = 10$
Local PCA in RM-MEDA	14 clusters for DTLZ problems with 7 objectives and 5 clusters for other problems

3.6 Results and Discussions

3.6.1 Results on High-dimensional Problems

Figures 3.4-3.15 show the box-plot in terms of IGD and NR for 10 independent runs, 20 and 200 decision variables for ZDT1, ZDT2, ZDT3, ZDT6, DTLZ1, and DTLZ3, respectively. For ZDT problems with 20 decision variables, it is observed that all the algorithms perform equally well and produce solutions that are very close to the global Pareto optimal front as indicated by near-zero IGD values. Among the algorithms, RM-MEDA shows the worst performance. In DTLZ problems with 20 decision variables, NSGA-II performs slightly better than REDA and the performance of MOUMDA and RM-MEDA are the worst. The poor performance of RM-MEDA

and MOUMDA may be attributed to the fact that, neither of the algorithms directly uses location information of the selected solutions in exploiting the new solutions. Even though REDA does not make use of location information, it performs better than MOUMDA and RM-MEDA. A possible explanation could be due to the learning capability of the network. The network can learn the global distribution of the solutions by allowing the survival of some individuals with large energy values; consequently, improving the exploitation capability of the algorithm. The difficulty of the DTLZ1 and DTLZ3 problems is caused by the many local Pareto optimal fronts. It seems that MOEDAs (REDA, MOUMDA and RM-MEDA), which model the global probability of the selected individuals, may not be able to escape from local optima. On the other hand, NSGA-II, with stochastic recombination and the usage of location information for generating new solutions, may have a greater chance of escaping from local optima. Thus, NSGA-II generally evolves good result in both the test problems. However, all the algorithms fail to converge to the global Pareto optimal front in both DTLZ problems. In terms of NR, REDA produces more non-dominated solutions than the other algorithms for ZDT1, ZDT2, and ZDT6, while NSGA-II produces the most non-dominated solutions for ZDT3, DTLZ1, and DTLZ3.

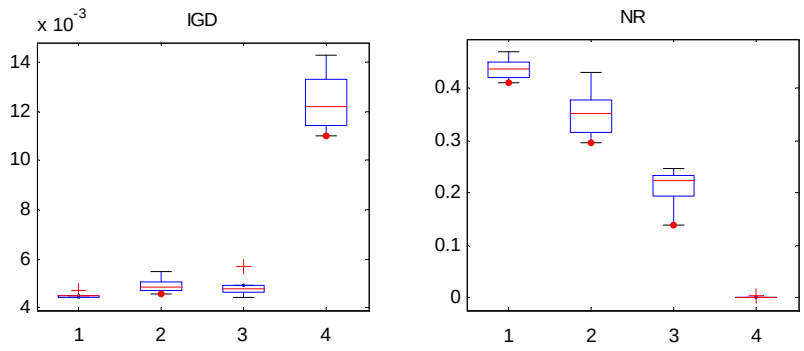


Figure 3.4: Performance metric of IGD and NR for ZDT1 with 20 decision variables

When the number of decision variables is increased 10 times to 200, it is observed that REDA outperforms the other algorithms in all the test functions, achieving a lower value of IGD and producing more non-dominated solutions. The increase in number of decision variables in-

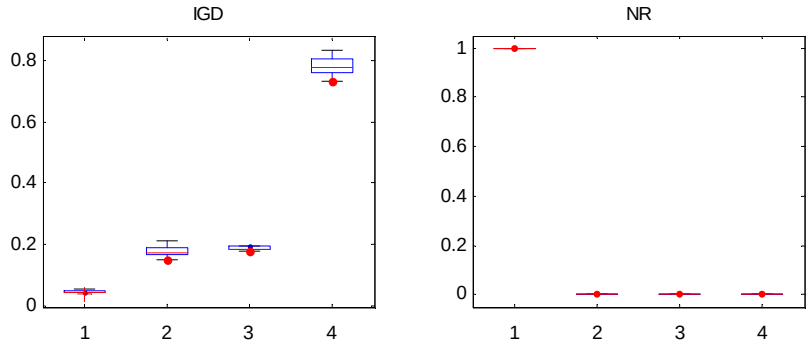


Figure 3.5: Performance metric of IGD and NR for ZDT1 with 200 decision variables

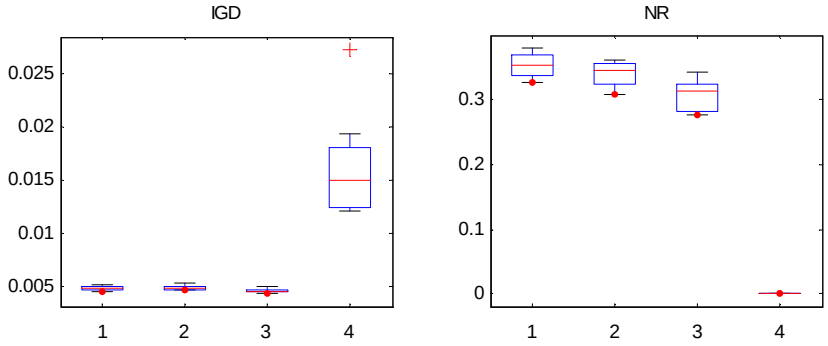


Figure 3.6: Performance metric of IGD and NR for ZDT2 with 20 decision variables

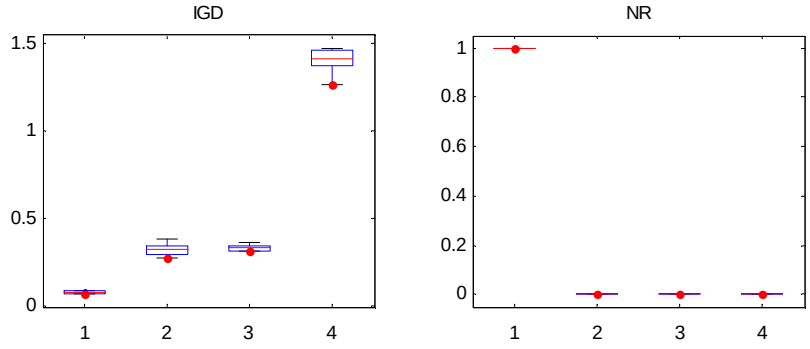


Figure 3.7: Performance metric of IGD and NR for ZDT2 with 200 decision variables

creases the size of the search space, and thus increases the complexity of the problems. The stochastic behaviour of NSGA-II prevents the algorithm from evolving a good set of solutions since the number of possible combinations towards optimality is increased. The univariate factorization in MOUMDA can only build the independent probability model which does not give extra information to guide the search. For RM-MEDA, the algorithm performs poorly in all the test problems. This may be attributed to the fact that RM-MEDA is unable to exploit the regularity of

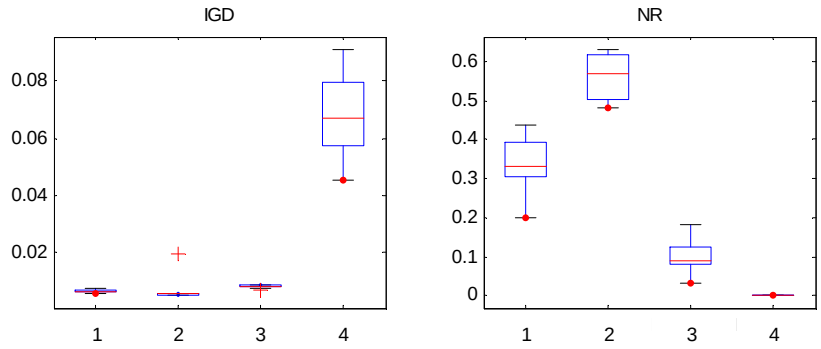


Figure 3.8: Performance metric of IGD and NR for ZDT3 with 20 decision variables

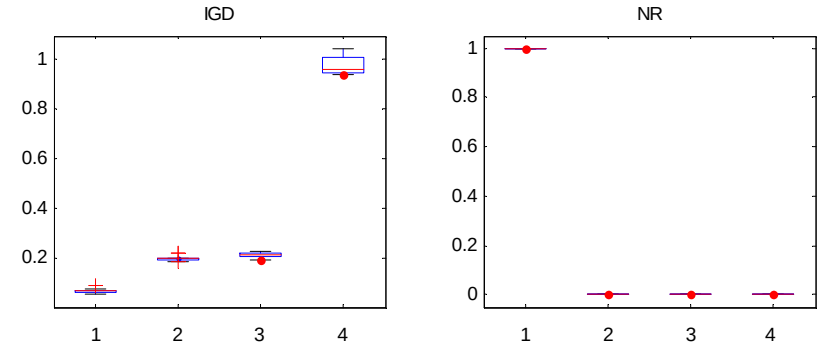


Figure 3.9: Performance metric of IGD and NR for ZDT3 with 200 decision variables

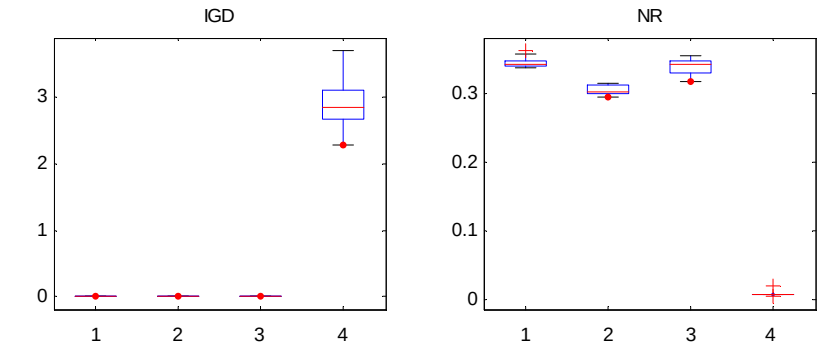


Figure 3.10: Performance metric of IGD and NR for ZDT6 with 20 decision variables

the Pareto set, since a large number of variables may weaken the regularity of the solutions during the initial generations. Even though artificial linkage dependencies within the variables were not introduced, it is believed that there may be certain implicit relationships between the decision variables, which are hard to measure and predict. The incorporation of dependency information to predict the true probability distribution allows REDA to obtain better results compared to the other algorithms.

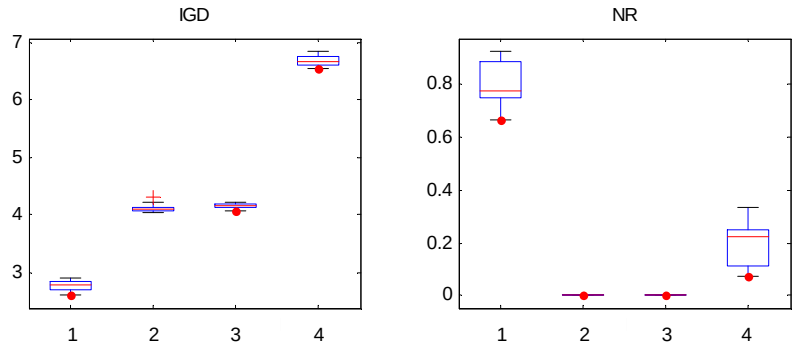


Figure 3.11: Performance metric of IGD and NR for ZDT6 with 200 decision variables

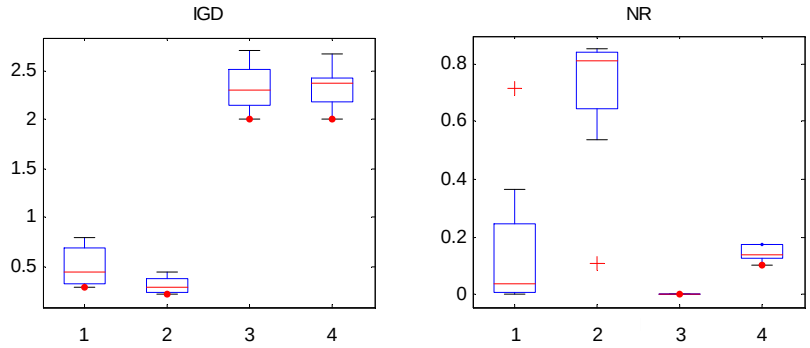


Figure 3.12: Performance metric of IGD and NR for DTLZ1 with 20 decision variables

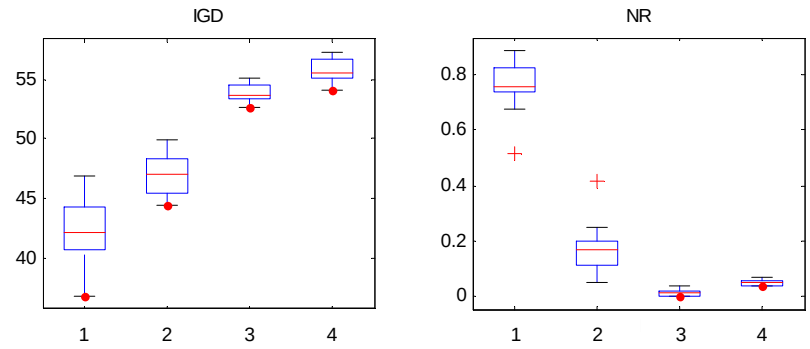


Figure 3.13: Performance metric of IGD and NR for DTLZ1 with 200 decision variables

In order to observe the behaviours of the algorithms for problems with a very large number of decision variables, performance metric of IGD for ZDT1, ZDT3, DTLZ1, and DTLZ3 with different number of decision variables are plotted in Figure 3.16. For ZDT1 and ZDT3, the number of variables spans from 100 to 1000. For DTLZ1 and DTLZ3, the number of variables spans from 100 to 500. Only the results of REDA, NSGA-II, and MOUMDA are plotted. RM-MEDA is left out of the comparative plots as it fails to meet the timing constraint when there

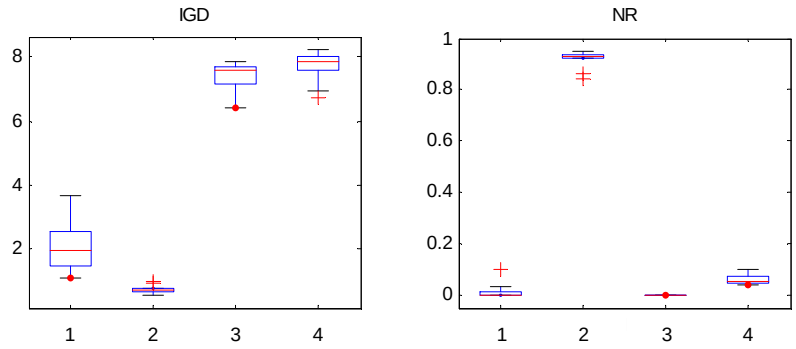


Figure 3.14: Performance metric of IGD and NR for DTLZ3 with 20 decision variables

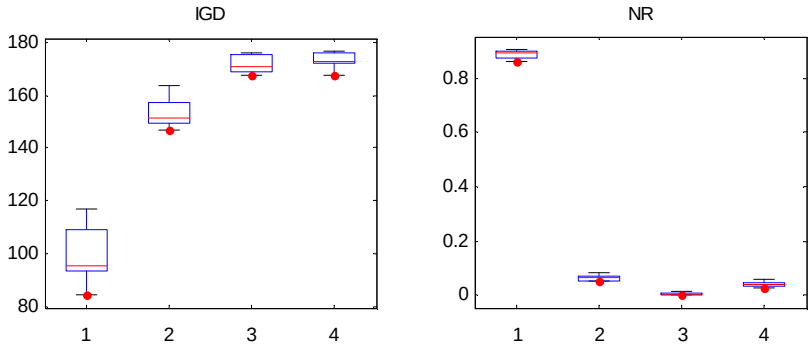


Figure 3.15: Performance metric of IGD and NR for DTLZ3 with 200 decision variables

are a very large number of decision variables. From the figure, it can be observed that REDA performs very well in ZDT1 and ZDT3. REDA also outperforms the other algorithms in DTLZ1 and DTLZ3. However, its performance deteriorates with the increase in the number of decision variables due to the increasing ease at which the algorithm gets trapped in local optima. This is the weakness of the REDA (and the other algorithms too).

3.6.2 Results on Many-objective Problems

DTLZ1, DTLZ2, DTLZ3, and DTLZ7 are characterized by linear Pareto optimal front, spherical Pareto optimal front, many local optimal fronts, and disconnected Pareto optimal fronts, respectively. All of these test problems can be scaled to a larger number of objective functions. This may pose a greater challenge to the algorithms in their search for a high-dimension Pareto front. Figures 3.17-3.24 show the performance metrics of IGD and NR for DTLZ1, DTLZ2, DTLZ3, and DTLZ7 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives. The number of decision

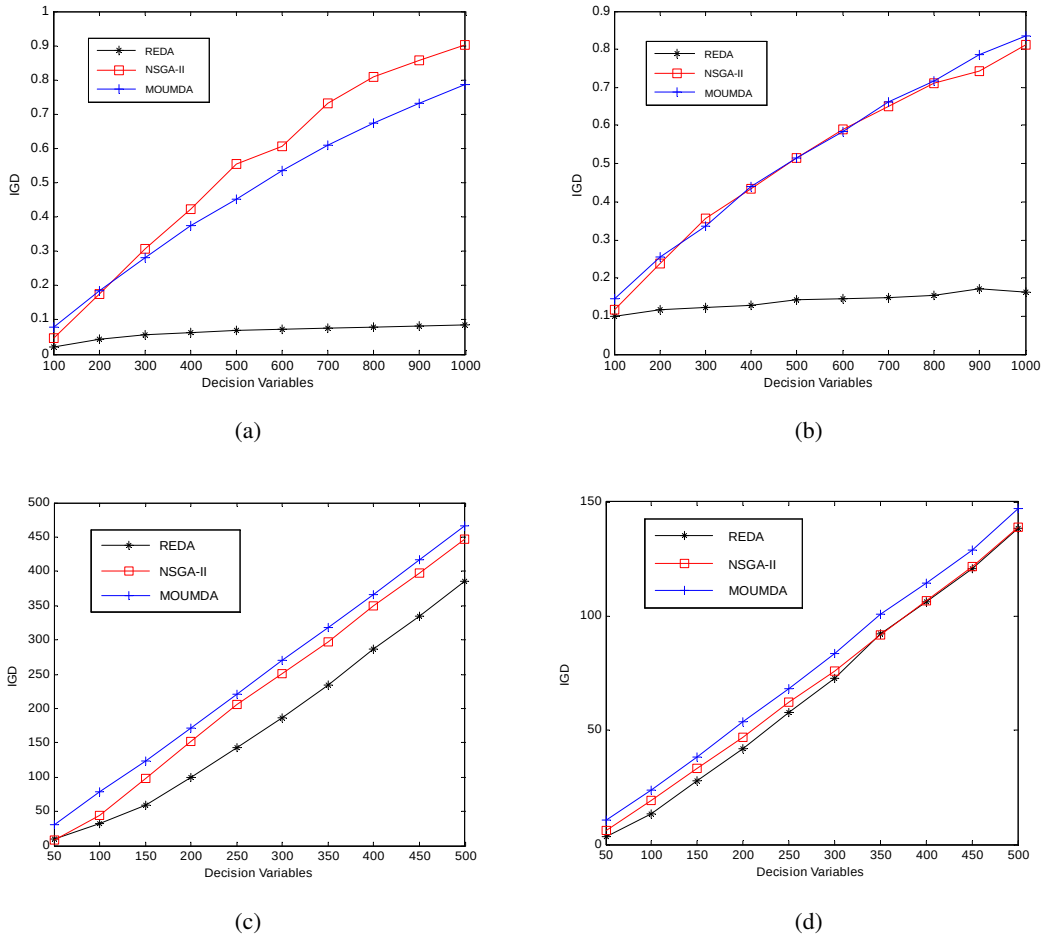


Figure 3.16: Performance metric of IGD versus the number of decision variables for (a) ZDT1, (b)ZDT3 (c)DTLZ1 and (d) DTLZ3

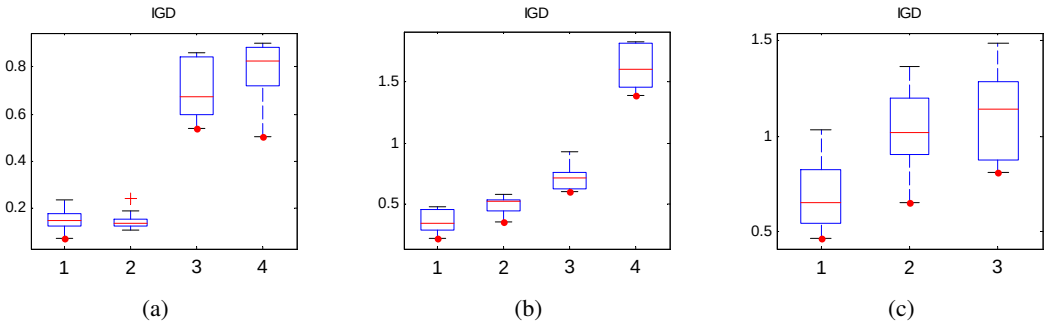
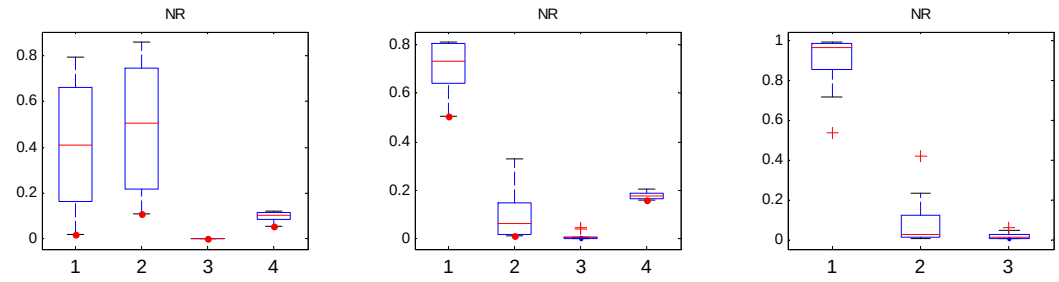


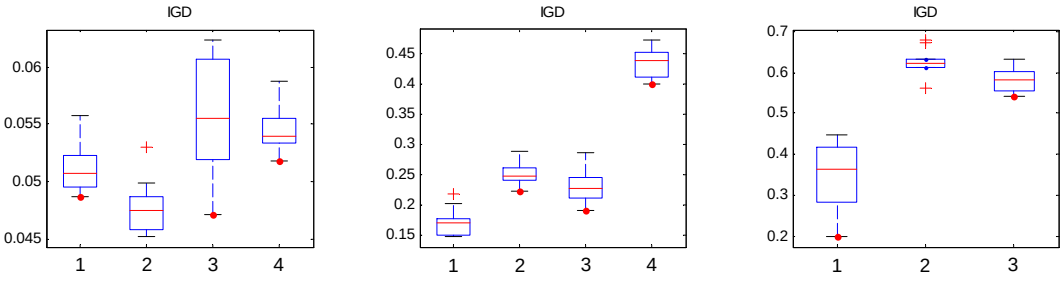
Figure 3.17: Performance metric of IGD for DTLZ1 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

variables (n) is fixed according to $n = m + K - 1$, where m is the number of objective functions and $K = 10$. Again, the results of RM-MEDA for problems with 7 objectives are not shown due to the timing constraint in evolving a final front. The population size is varied according to



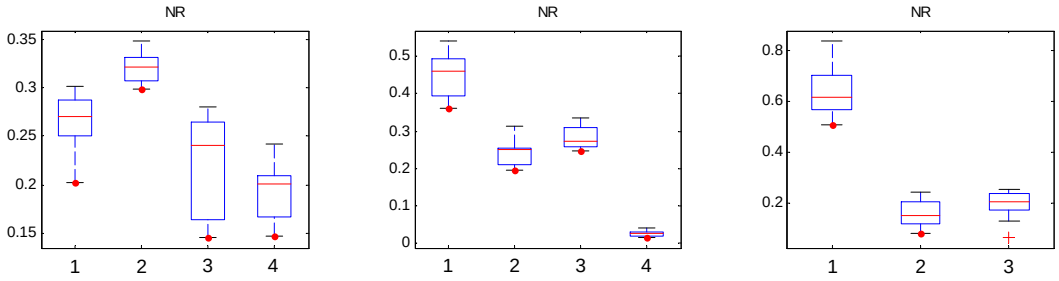
(a) (b) (c)

Figure 3.18: Performance metric of NR for DTLZ1 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives



(a) (b) (c)

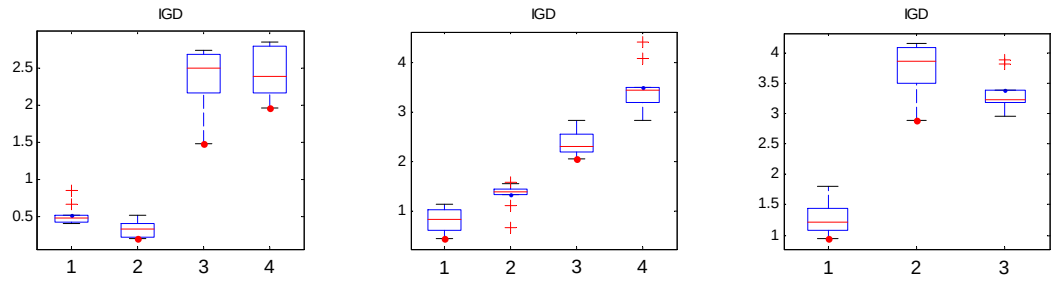
Figure 3.19: Performance metric of IGD for DTLZ2 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives



(a) (b) (c)

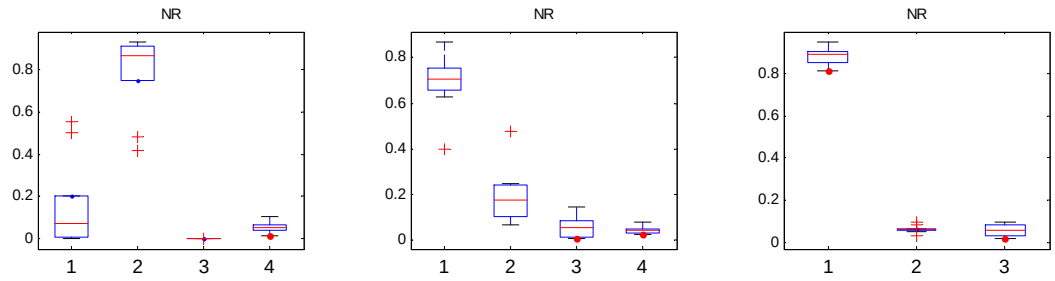
Figure 3.20: Performance metric of NR for DTLZ2 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

the number of objectives as indicated in Table 3.2. In the preliminary experiment, it is observed that evolution does not take place effectively when the population size is small as the algorithms fail to identify the fitter individuals. To deal with this problem, the population size was increased according to the number of objectives to provide the algorithms with a better chance of selecting the fitter solutions.



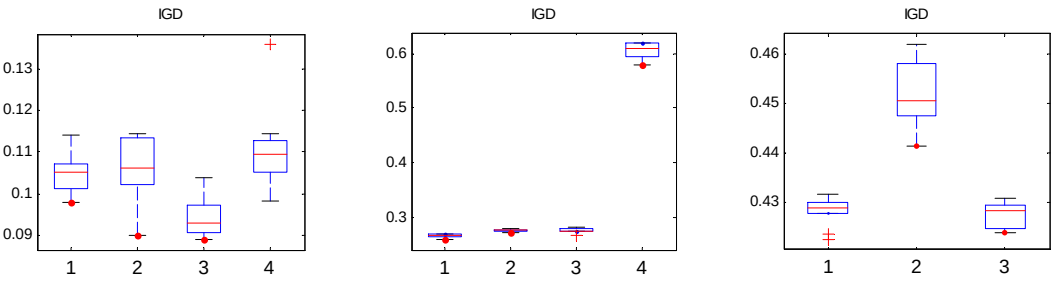
(a) (b) (c)

Figure 3.21: Performance metric of IGD for DTLZ3 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives



(a) (b) (c)

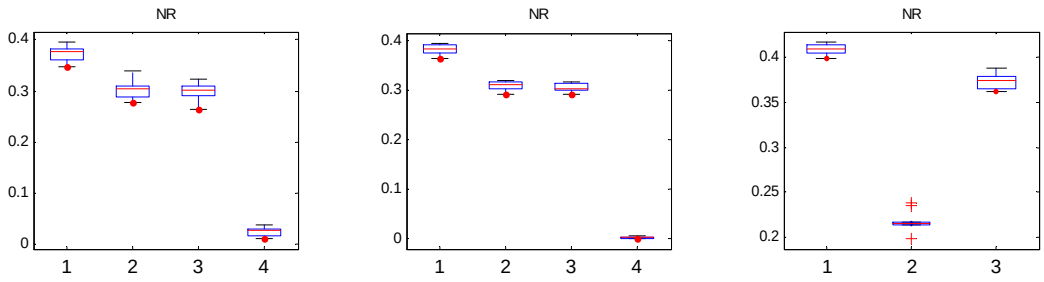
Figure 3.22: Performance metric of NR for DTLZ3 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objective



(a) (b) (c)

Figure 3.23: Performance metric of IGD for DTLZ7 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

It is observed that REDA and NSGA-II perform equally well for all the test instances with 3 objective functions, while MOUMDA and RM-MEDA perform worst in DTLZ1 and DTLZ3. However, MOUMDA performs better than the other algorithms in DTLZ7 (indicated by IGD value). When the number of objective functions is increased to 5 and 7, REDA seems to outperform the other algorithms, in terms of both IGD and NR, in all the test problems. Even though all



(a) (b) (c)

Figure 3.24: Performance metric of NR for DTLZ7 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

the algorithms fail to converge to the global Pareto optimal front in all the test instances (as indicated by high values of IGD), the solutions generated by REDA seems closer to the true Pareto optimal front. The superior performance of REDA may be due to the incorporation of linkage information in driving the search. This information is learnt by the network and is clamped into the probability distribution before the sampling takes place. Some flexibility is given to the algorithm in exploring the search space by allowing the training to stop before the energy reaches the minimum. The good performance of REDA in these test instances supports the claim that REDA scales well with the number of objective functions compared to the other algorithms.

3.6.3 Effects of Population Sizing on Optimization Performance

Test problems of ZDT1 with 100 decision variables (ZDT1-100) and DTLZ1 with 5 objectives and 14 decision variables (DTLZ1-5-14) are used in this experiment. Simulations are conducted on the REDA and NSGA-II with a population size of 20, 50, 100, 200, and 500, and the results are tabulated in Table 3.3. The simulations stop at 20,000 fitness evaluations for ZDT1 and 100,000 fitness evaluations for DTLZ1. According to the table, NSGA-II prefers a larger number of generations rather than a larger population size for better performance, while REDA needs a larger population size rather than a larger number of generations for better performance, in both two-objective and five-objective problems. This could be due to the fact that REDA can model the distribution of the solutions more accurately when the input data is large. When the training

data is small, the distribution of the fitter solutions may not represent the optimal distribution in the particular generation. Thus, it affects the sampling capability of the algorithm in producing fitter solutions. As long as the distribution of the fitter solutions is being correctly modelled, fitter offspring can be generated. Therefore, a large population size is crucial to the success of the REDA.

Table 3.3: IGD metric for ZDT1 and DTLZ1 with different population size

	REDA					
	ZDT-100			DTLZ1-5-14		
Population Size	Mean	Std Dev	Median	Mean	Std Dev	Median
20	0.2957	0.0180	0.2925	1.3435	0.3282	1.3575
50	0.0512	0.0056	0.0526	1.2838	0.1906	1.2696
100	0.0201	0.0041	0.0190	1.3009	0.2658	1.2378
200	0.0107	**0.0021**	**0.0104**	0.6553	0.2225	0.5864
500	**0.0105**	0.0023	**0.0104**	**0.3547**	**0.0896**	**0.3404**
	NSGA-II					
	ZDT-100			DTLZ1-5-14		
Population Size	Mean	Std Dev	Median	Mean	Std Dev	Median
20	**0.0388**	0.0077	**0.0385**	0.4515	0.1106	0.4657
50	0.0468	0.0066	0.0482	**0.4164**	0.1173	**0.3985**
100	0.0464	0.0067	0.0452	0.5263	0.1016	0.5351
200	0.0613	**0.0050**	0.0610	0.5843	0.1615	0.5455
500	0.2679	0.0073	0.2699	0.6893	**0.0975**	0.7090

3.6.4 Effects of Clustering on Optimization Performance

The effect of different number of clusters on optimization performance is studied and the results are tabulated in Table 3.4. According to the table, clustering is unnecessary for ZDT1, but is of utmost importance for DTLZ1. Clustering is important for problems with solutions that are hard to represent by a probability distribution. Since the coverage solutions of ZDT1 are quite evenly distributed, the clustering would only reduce the number of data for modelling. Clustering is important for high-dimensional problems with complex Pareto optimal fronts to cluster the representative solutions into different regions for modelling purpose. Therefore, this chapter concludes that the necessity of clustering is problem dependent.

3.6.5 Effects of Network Stability on Optimization Performance

The stability of the network is particularly determined by the number of hidden units. A large number of hidden units may give the network extra capacity to learn the distribution of the input

Table 3.4: IGD metric for ZDT1 and DTLZ1 with different number of clusters

	ZDT-100			DTLZ1-5-14		
Cluster	Mean	Std Dev	Median	Mean	Std Dev	Median
1	**0.0201**	**0.0041**	**0.0190**	1.0044	0.2966	1.0435
3	0.0530	0.0071	0.0506	0.3858	0.1046	0.3855
5	0.0981	0.0107	0.0952	0.3855	0.1513	0.3809
7	0.1422	0.0193	0.1407	0.3547	0.0896	0.3404
9	0.2239	0.0244	0.2347	0.3420	0.1154	0.3486
11	0.2433	0.0442	0.2414	0.3235	**0.0798**	0.3019
13	0.4040	0.0663	0.4117	**0.3046**	0.0873	**0.2801**
15	0.4903	0.0490	0.4902	0.4098	0.0971	0.4070

Table 3.5: IGD metric for ZDT1 and DTLZ1 with different number of hidden units

	ZDT-100			DTLZ1-5-14		
Hidden unit	Mean	Std Dev	Median	Mean	Std Dev	Median
1	0.4296	0.0382	0.4248	0.5068	0.1797	0.5189
5	0.1599	0.0114	0.1586	0.3547	0.0896	0.3404
10	0.0508	0.0081	0.0355	**0.2941**	0.0899	0.2948
20	**0.0201**	**0.0041**	**0.0190**	0.2974	0.0745	**0.2778**
50	0.0298	0.0049	0.0285	0.3599	**0.0544**	0.3434

data. Thus, the network can converge to lower energy equilibrium (more stable). In this section, the performance of REDA with different number of hidden units is examined and the results are presented in Table 3.5. From the table, it is observed that the smallest number of hidden unit (1 unit) and the largest number of hidden units (50 units) gave the worst performance compared to the intermediate settings (5-20 units). The function of hidden neurons is to determine the learning capacity of the network in deriving the probability distribution of the solutions. Even though a large number of hidden units may reduce the final training error, empirical results show that an intermediate number of hidden units would give better performance. This observation suggests that extensive modelling of the distribution for a selected population at a particular generation is unnecessary. This is because the exact distribution of the selected population at the current generation may not represent the real optimal distribution. Furthermore, it may be beneficial to give flexibility to the algorithm (allowing large energy value) to explore the search space. On the other hand, having too few hidden neurons may yield large training errors, resulting in a model that may deviate from the true distribution, and consequently, poor algorithmic performance.

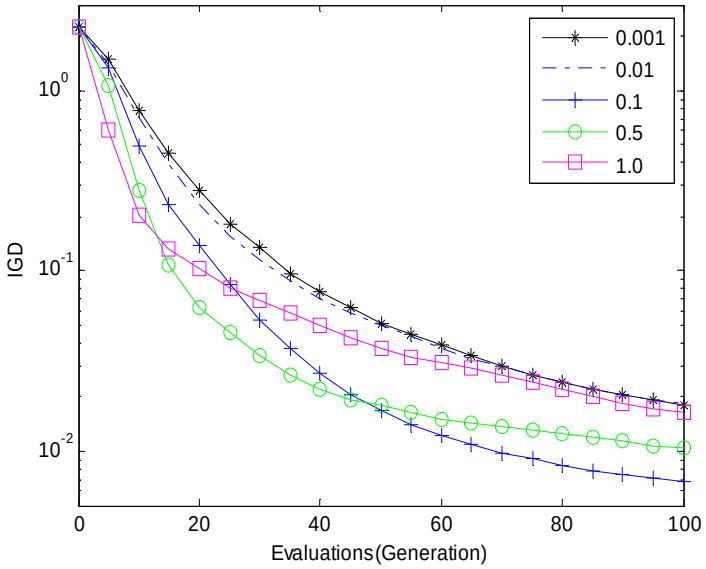


Figure 3.25: Performance metric of IGD for REDA with different settings of learning rate in ZDT1

3.6.6 Effects of Learning Rate on Optimization Performance

The learning rate in RBM will determine how fast and how proper the network is trained. Lower learning rates may slow down the training process while larger ones may overtrain the network. Therefore, the setting of learning rate is crucial for training the RBM. Figure 3.25 plots the IGD trace for the REDA using different learning rates in solving ZDT1. It is observed that smaller learning rates (0.001 and 0.01) may slow down the speed of convergence and is unable to evolve fitter solutions by the final generation. On the other hand, larger learning rates (0.5 and 1.0) may speed up the convergence during the early generations but the performance slows down after a number of generations and eventually, the convergence stops before optimality is reached. The best learning rate is around 0.1 where the algorithm has faster convergence speed and at the same time, is able to find better final solutions.

3.6.7 Computational Time and Convergence Speed Analysis

Most of the probabilistic modelling techniques for learning the linkage dependencies of the solutions incur additional computational cost and time. In RBM, the most time consuming part

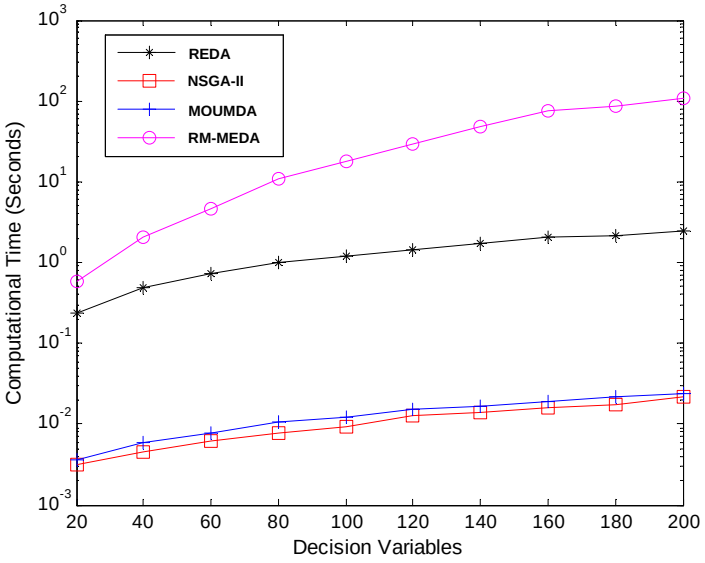
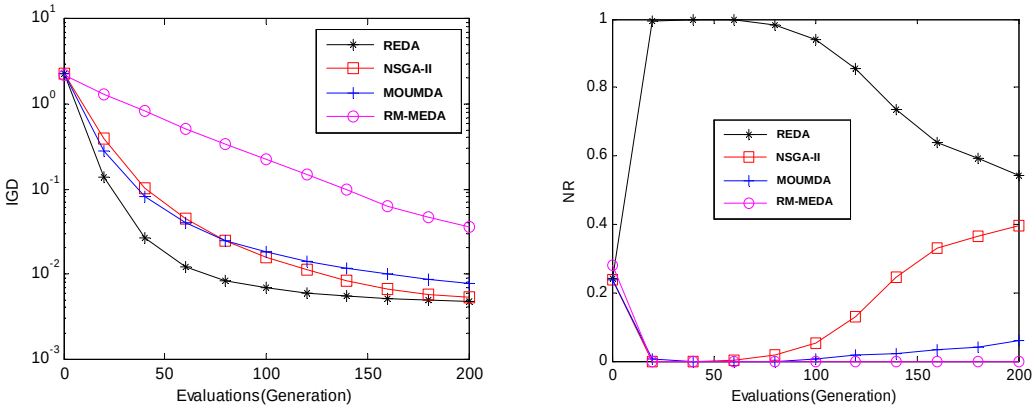


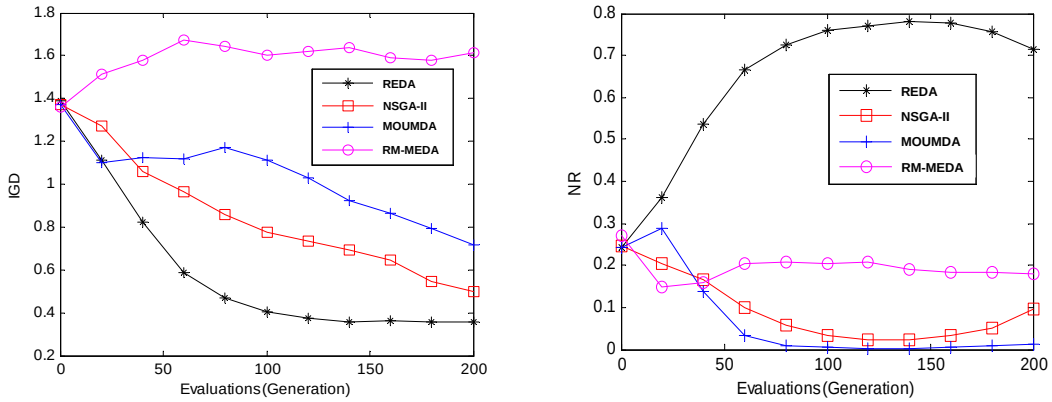
Figure 3.26: Computational time for various algorithms in ZDT1 with different number of decision variables



(a) (b)

Figure 3.27: Performance traces of (a) IGD and (b) NR for ZDT1 with 30 decision variables

is the network training. Training is conducted at each generation and stops when the maximum number of training epochs is reached. This training process is more complicated than the genetic operators in standard MOEAs, and thus incurs additional simulation time. The computational time for solving ZDT1 with different number of decision variables using the algorithms with 200 generations and a population size of 100 is plotted in Figure 3.26. From the figure, it is clear that MOUMDA and NSGA-II take less computational time than REDA and RM-MEDA, with RM-MEDA being the most time consuming algorithm. Figures 3.27 and 3.28 show the performance



(a) (b)

Figure 3.28: Performance traces of (a) IGD and (b) NR for DTLZ1 with five objectives and 14 decision variables

trace of IGD and NR for ZDT1 and DTLZ1. Even though REDA may spend more simulation time, it has a faster convergence rate compared to MOUMDA, NSGA-II, and RM-MEDA. This is one of the strengths of REDA, especially when dealing with real-world applications where the fitness evaluations are more computationally expensive.

3.7 Summary

In this chapter, a new EDA based on RBM in the context of multi-objective optimization has been presented. Unlike most EDAs, which model the probability distribution by statistical methods, REDA models the probability distribution through unsupervised learning and stores the statistical information in the network's weights and biases. This information is subsequently returned as a probability distribution of the solutions, where the distribution is proportional to the energy function of the network. Learning characteristic is one of the main features of REDA, which gives flexibility to the algorithm in modelling hard and complex distributions. The k-mean clustering has been employed to group the population, in objective space, into smaller clusters. It has been shown to improve the performance of REDA when solving problems with high-dimensional Pareto fronts. The effects of population size, number of hidden units, number of training epochs, number of clusters, computational time, and learning rate have been experimentally studied. It

is observed that REDA has the fastest convergence rate but is sensitive to the number of hidden units and training epochs.

Even though REDA has shown promising results in solving scalable problems, it still suffers several flaws which require further investigation. Firstly, univariate sampling in REDA may limit its ability to generate new solutions. This is because univariate sampling does not consider the correlation between the decision variables when performing the sampling. A more sophisticated sampling mechanism that is able to take into account the explicit multivariate dependencies between the decision variables may enhance the search capability of the REDA. Secondly, REDA fails to converge to the global Pareto optimal front in problems with many local optima. This is because REDA or MOEDA in general will model the probability distribution of the solutions even though they are trapped at local optima, and subsequently use the constructed probabilistic information as a reference model to produce offspring. Hybridization with local search may be one of the approaches in dealing with problems with many local optima. Thirdly, REDA or MOEDA in general is sensitive to bias. This is because the modelling in EDAs only estimates the probability distribution of the current best solutions. In other words, only global information is used. Whenever the maintained solutions are biased towards certain search regions, EDAs may consider the maintained solutions are the promising one, thus, construct their probability distribution accordingly. Therefore, it is necessary to enhance the diversity preservation of REDA especially the ability to produce a set of diverse solutions. This can be achieved by combining EDA with other search algorithms which use location information in producing offspring, including genetic algorithm, differential evolution, particle swarm optimization algorithm, or any other algorithms with similar features. Thus, more researches should be carried out to study the model building approach, sampling technique, hybridization, and utilization of location information in order for REDA to solve problems with complicated Pareto fronts effectively.

Chapter 4

An Energy-based Sampling Mechanism for REDA

This chapter examines the sampling techniques of REDA. The behaviours of the simple probability sampling technique, in terms of energy levels, are rigorously investigated. Then, a sampling mechanism with two different selection schemes that exploits the energy information of the solutions in a trained network is proposed to improve the search capability of the algorithm. Several benchmark problems, together with their artificially introduced linkage dependency versions, are used to examine the efficiency of the proposed energy-based sampling mechanism. Empirical studies show that the new sampling mechanism gives promising results in terms of inverted generational distance and non-dominance ratio.

4.1 Background

EDAs, also known as iterated density estimation algorithms (IDEAs) [125] and probabilistic model building genetic algorithms (PMBGAs) [126], are a new computing paradigm in the field of evolutionary computation (EC). In the previous chapter, a neural-based EDA for multi-objective optimization (REDA) has been devised. While the REDA has been shown to be able

to solve high-dimensional problems with a large number of decision variables and objective functions, its performance, and those of MOEDAs in general, is particularly dependent on the probabilistic modelling and sampling techniques used. Over the past few years, many modelling approaches, including Bayesian tree, decision tree, principle component analysis, have been studied. On the other hand, sampling techniques for MOEDAs are less studied and developed.

This chapter studies the sampling procedure in REDA. Even though REDA can model the multivariate dependencies between the decision variables, the final probability distribution is clamped into a marginal distribution of the decision variables. Subsequently, sampling is simply carried out based on the marginal distribution to produce new solutions. This feature may reduce the efficiency of the algorithm in exploring the space of potential solutions in cases where the number of decision variables is high, and when the decision variables have strong linkage dependencies. In order to fully utilize the information provided by the trained network (energy value), the characteristics of the sampled solutions, in terms of the energy level, are examined. This is important since the energy value is derived from the dependency information of the network. This investigation leads us to propose a new sampling mechanism that makes use of the energy information. Two selection schemes for the proposed energy-based sampling mechanism are created. The efficiency of the energy-based sampling mechanism is rigorously examined under eight benchmark test instances and three variants with artificially introduced linkage dependencies [110,127]. Finally, performance indicators, such as inverted generational distance (IGD) and non-dominance ratio (NR), are used to evaluate the performance of the energy-based sampling mechanism. The empirical studies show that the proposed energy-based sampling mechanism gives promising results in terms of IGD, NR, and the rate of convergence.

The rest of this chapter is organized as follows. Section 4.2 presents an investigation on the probabilistic modelling and sampling techniques of REDA. Section 4.3 introduces the energy-based sampling mechanism. Experimental setups and test functions are illustrated in Section 4.4. Results and further analyses are outlined in Section 4.5 and Section 4.6 concludes this chapter.

4.2 Sampling Investigation

Much research has been conducted to design new EDAs based on various machine learning approaches or probabilistic graphical models [106, 128–130]. However, not much work has been conducted to analyze the structures and the behaviours of the probabilistic modelling and sampling techniques [131–133]. This knowledge can be used to design practical theoretical models, problem-based operators, and enhancement approaches [134]. In this section, we carry out an investigation on how well the states are being reconstructed by the RBM over different training epochs, how to effectively train an RBM, and what can be elucidated from the energy values of an RBM.

4.2.1 State Reconstruction in an RBM

In an RBM, the neurons between two layers are fully connected via weighted synaptic connections. However, there is no intra-layer connection. These weight connections are used by the neurons to communicate their activations to one another. During the learning process, the activation states of the network comprise of a Boltzmann probability distribution. The quality of training of the network corresponds directly to the effectiveness at which the algorithm learns the probability distribution. Moreover, the network captures the energy of the data to elucidate the relationships between the decision variables. This distribution-based model allows the RBM to globally learn the probability distribution of the decision variables by considering the interdependencies of the data.

In order to understand what is reconstructed by the RBM model and how well the network can construct the data points, the distributions of the input and reconstructed data points in the decision space for POL problem [135] are presented in Figure 4.1. Dark circles are the input data and blank circles are the reconstructed data. The POL problem is chosen because it only consists of two decision variables, which allows the decision frontier to be more easily visualized. The training error at generation g is measured by the Euclidean distance between the real data points

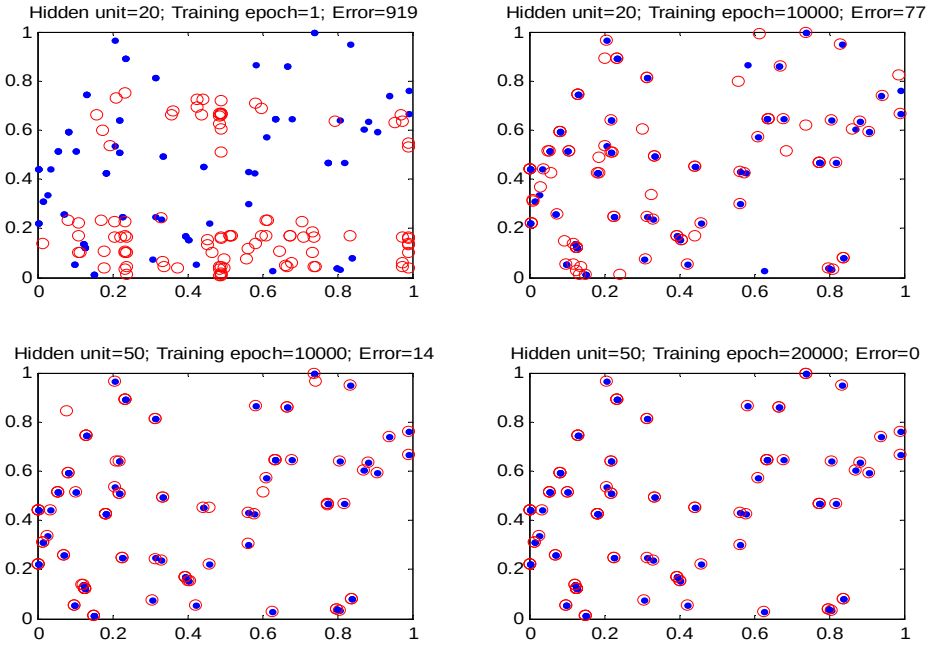


Figure 4.1: Distribution plots of the input data points (dark circles) and reconstructed data points (blank circles) generated by an RBM. Different number of hidden units and training epochs are used, which result in different training errors and thus different sets of reconstructed data.

(input data) and the reconstructed data points calculated according to the following equation:

$$Error_g = \sum_{i=0}^{N} \sum_{j=0}^{n} (x_{i,j} - x'_{i,j})^2 \tag{4.1}$$

where N is the population size, n is the number of decision variables, $x_{i,j}$ is the real data point, and $x'_{i,j}$ is the reconstructed data point. From Figure 4.1, it is observed that with light training (upper left subfigure), the training error is 919, indicating that the algorithm may not correctly model the distribution of the data. The distribution of the reconstructed data points is near to one obtained by random initialization. The reconstruction is improved with further training (upper right subfigure) which reduces the training error to 77.

When the number of hidden neurons (50 units) and training epochs (20,000 epochs) are sufficiently large, the network succeeded in reconstructing the real data points (lower right sub-

figure). In this case, the training error is 0. Therefore, an RBM is able to model the exact data points and the distribution of the solutions is captured in the synaptic weights of the network. This observation that a sufficient number of hidden units in the network would guarantee improvement in the training error has been proven mathematically in [136]. In other words, the network can represent any discrete distribution exactly when the number of hidden units is very large.

4.2.2 Change in Energy Function over Generations

The weight update process in an RBM requires calculating the gradient of log-likelihood of the input data. The gradient is minimal when the reconstructed data is exactly similar to the input stimuli. Contrastive divergence training [122] aims to obtain the weights of the network that minimize the energy level and training error of the network. The primary understanding is that the minimal energy level and training error can be achieved when sufficient number of hidden units and training epochs are applied. This is because the learning capability of the network is determined by the number of hidden units. A larger number of hidden units gives extra flexibility for the network to model the global distribution of the input stimuli, and thus could yield better convergence. On the other hand, contrastive divergence training will require a large number of training epochs to train the network well.

When the RBM is modelled as an EDA, another factor that can reduce the energy level and training error is the number of generations of an optimization process. Over generations, the training error and energy level of the network are reduced. This is shown in Figure 4.2, where H20 E50 represents an RBM setting consisting of 20 hidden units and 50 training epochs. The data is obtained by running REDA on POL and KUR [137] problems. From Figure 4.2, it is observed that the energy level and training error are reduced over generations. This observation suggests that extensive training is unnecessary during the earlier generations because the network more accurately models the distribution of the solutions towards the end of the evolution. This

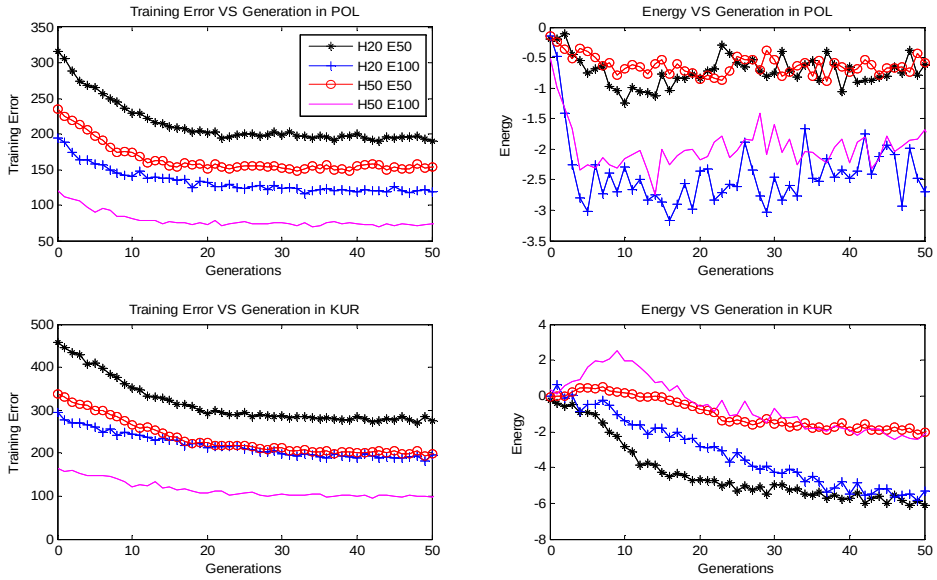


Figure 4.2: Training error and energy value versus generation produced by an RBM for different number of hidden units and training epochs

is most likely due to the reduction in the size of more promising search space when the search converges to near optimal points. By taking this into consideration, the computational time of the algorithm can be improved by eliminating unnecessary training of the network in each generation.

The energy values are also reduced over generation. However, the decrement in the training error does not proportional to the decrement in the energy equilibrium of the network (the upper figures). The network's energy equilibrium is dependent on the input and hidden states. This complex relationship prevents training epochs and hidden units from directly influencing the energy equilibrium of the network. This may be caused by the fact that certain choices of number of hidden units and training epochs may cause the network to be over trained or trapped in local optima. Thus, the setting of the hidden units should be carefully chosen. In the previous chapter, hidden units of 5-20 are suggested.

4.2.3 What Can be Elucidated from the Energy Values of an RBM

In EDAs, the two main mechanisms that determine the success of the algorithms are probabilistic model construction and sampling technique. The core purpose of the probabilistic modeling is

to learn the probability distribution of the candidate solutions by considering the dependencies between the decision variables. By using the probability density of the known solutions, the probability distribution of the unknown solutions can be studied. In EDAs, the parents are the known solutions while the offspring are the unknown solutions. If the characteristics of the offspring solutions can be predicted, this additional information can be taken into consideration during the optimization process.

In an RBM, the energy-based model captures the probability distribution of the parent solutions set by associating a scalar energy value from the network to each solution. Over the training process and over the evolutionary process, the energy is reduced to a certain level of thermal equilibrium. Thus, we can elucidate that the solutions that located inside the boundary regions of the parent solutions have a lower energy value. On the other hand, the solutions those are located outside of the boundary regions of the parent solutions may have a higher energy level. In pattern recognition, a lower energy level suggests that a test sample is more likely to belong to a certain class of patterns. However, this is not the case in EDAs as a lower energy level does not mean that the solutions are fitter, and vice versa. Figure 4.3 explains this claim. Figure 4.3 shows a set of input data (parent solutions) which was modeled by an RBM and a set of sampled data (offspring solutions) which was sampled from an RBM in an objective space with two objective functions. The solutions located inside the modelled boundary may have a lower energy value and the solutions located outside of the modelled region may have a higher energy level. If a selection scheme only selects solutions with a lower energy, the exploitation can be enhanced; however, the exploration is poor since the search only focuses on the boundary regions that have been modelled by the RBM. On the other hand, if the selection scheme only selects solutions with a higher energy level, the exploration can be enhanced. However, the exploitation is poor. Thus, a selection scheme should select both solutions with lower and higher energy levels. The proposed energy-based sampling mechanism in this chapter is based on this observation.

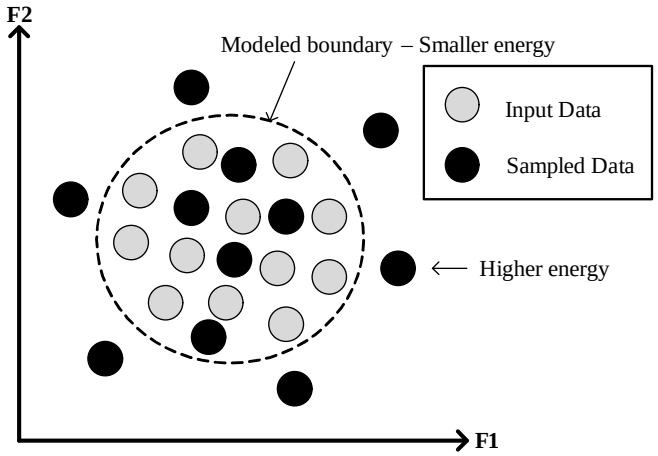


Figure 4.3: Training error and energy value versus generation produced by an RBM for different number of hidden units and training epochs

4.3 An Energy-based Sampling Technique

One of the main characteristics of REDA is its capability to learn the multivariate dependencies between the decision variables. This information is stored in the synaptic weights and biases of the network. The final probability distribution is constructed by clamping this information into the marginal probability of each decision variable or input unit in the network. The offspring for the following generation are subsequently sampled from the constructed probabilistic model. The simple sampling technique applied in REDA may, however, limit the production of appropriate solutions if the decision variables are highly correlated or have a high dimension. This is because, during sampling, marginal probability distribution considers the distribution of the particular decision variable but not the correlation between the decision variables. As a result, the sampled solutions have difficulties following the correlated distribution. One way to tackle this problem is to sample an infinite number of solutions. This may increase the number of possible combinations of the solutions and thus increase the chance of producing fitter individuals. However, sampling of an infinitely large number of solutions may lead to an increase in the number of fitness evaluations and computational time. It is known that some real-world problems are very time consuming and such an algorithm would not be practical.

To deal with the aforementioned problem, the energy value is taken into consideration.

Firstly, $N \times M$ solutions are generated. Then, the energy value will serve as the main criterion for forming new N solutions from the alleles of the $N \times M$ solutions, where $M > 1$ is a multiplier. A lower energy level implies that the solution is in a more stable state while a higher energy level means that the solution is not in energy equilibrium. The energy-based sampling mechanism will, therefore, prefer the alleles of solutions with lower energy levels.

As probabilistic modelling only models the previous best topology, the solutions that are located inside the modelled topology are stable (lower energy level) in terms of energy equilibrium and are generally fit. On the other hand, the solutions outside the modelled topology (higher energy level) may be considered unstable but not unfit (as observed in Section 4.2.3). This means that the solutions with higher energy levels may be the promising solutions that are not modelled by the network and thus will be worth preserving to the next generation. Therefore, it is required to give the algorithm the flexibility of choosing the alleles of solutions with high energy levels in order to achieve a more explorative search.

4.3.1 A General Framework of Energy-based Sampling Mechanism

Based on the above argument, the energy-based sampling mechanism is proposed (Figure 4.4). Firstly, $N \times M$ solutions are sampled according to equation (3.16) and stored in Z, which is then used as input for the RBM. Then, the corresponding states for each hidden unit h_j are constructed according to equation (3.5). Subsequently, the energy value for each of the solutions in Z is computed according to equation (3.1). After which, the energy values along with their corresponding indices, are sorted in an increasing order. Finally, the selection procedure is applied to form new N solutions using the alleles of the $N \times M$ solutions. The selection procedure is crucial in forming the new solutions. Two selection procedures are proposed in this chapter.

Begin
%%Replace the eighth step of Figure 3.3
Given the marginal probability of each decision variable, $p(v_i)$
1. Sample $N \times M$ solutions according to equation (3.16) and store them in Z
2. Reconstruct the states for hidden units of all solutions produced in Step 1 using equation (3.5)
3. Compute the energy values for all solutions using equation (3.1)
4. Sort the energy values in an increasing order
5. Apply selection procedure to form new N individuals as the final offspring

End

Figure 4.4: Pseudo-code of the energy-based sampling mechanism

4.3.2 Uniform Selection Scheme

Under this scheme, the alleles of all the $N \times M$ solutions have equal chance of being selected. The process is shown in Figure 4.5, where $Pop(g)_{i,j}$ is the new i^{th} solution's j^{th} bit position at generation g and $Z_{RandA,j}$(g) is the sorted $RandA^{th}$ solution's j^{th} bit position at generation g. The new formed population using USS consists of the alleles of the solutions with uniformly distributed energy values. In other words, the number of low and high energy individuals' alleles is almost similar.

Begin
%%Implemented in the fifth step of the energy-based sampling
%%mechanism in Figure 4.7
For $\boldsymbol{i = 1:N}$
For $\boldsymbol{j = 1:n \times b}$
$RandA = uniform\ random\ value\ between[1, N \times M]$
$Pop(g)_{i,j} = Z_{RandA,j}\ (g)$
End For $\boldsymbol{j}$
End For $\boldsymbol{i}$
End

Figure 4.5: Pseudo-code of the uniform selection scheme (USS). n is the number of decision variables and b is the number of bits per variable.

4.3.3 Inverse Exponential Selection Scheme

Under this scheme, there is a higher probability of selecting the alleles of individuals with lower energy values. The pseudo-code for this scheme is presented in Figure 4.6, where $Pop(g)_{i,j}$ is the new i^{th} solution's j^{th} bit position at generation g and $Z_{RandC,j}$(g) is the sorted $RandC^{th}$ solution's j^{th} bit position at generation g. min and max determine the range of the random

values. In order to allow the flexibility of changing the range of the random values, α is added to the algorithm. For simplicity, min and max are permanently assigned values of 0.01 and 1.0, respectively. In this way, α is the only parameter that determines the probability at which an allele of an individual will be selected. The probabilities of selecting each allele of the individuals for different values of α are shown in Figure 4.7. In the figure, it is observed that a smaller value of α will result in a more even chance of selecting any alleles of the individuals. The probability of selecting the alleles of the individuals with lower energy values increases and the probability of selecting the alleles of the solutions with higher energy values decreases with the increase in the value of α. This scheme is designed based on the observation that solutions with lower energy values are located in the topology modelled by the previously selected population. These solutions are moderately fit. When the new formed population consists mostly of the alleles of such solutions, the overall probabilistic model will not produce any solutions that is far from promising regions. Furthermore, individuals with higher energy values located outside of the previously modelled region may be fit or unfit. Therefore, the alleles of these solutions have a lower probability of being selected in order to increase the exploration capability of the algorithm.

Begin
%%Implemented in the fifth step of the energy-based sampling
%% mechanism in Figure 4.7
For $\boldsymbol{i = 1 : N}$
For $\boldsymbol{j = 1 : n \times b}$

$$RandB = \ uniform\ random\ value\ between\ (\alpha \times min, \alpha \times max]$$

$$RandC = \left(Exp\ (RandB) - Exp\ (\alpha \times min)\right) \times \frac{N \times M}{Exp\ (\alpha \times max\ ) - Exp\ (\alpha \times min\ )}$$

$$Pop(t)_{i,j} = z_{[RandC\ ],j}\ (t)$$

End For $\boldsymbol{j}$
End For $\boldsymbol{i}$
End

Figure 4.6: Pseudo-code of the inverse exponential selection scheme (IESS). n is the number of decision variables and b is the number of bits per variable.

The overall process of the selection scheme is shown in Figure 4.8. The $N \times M$ individuals

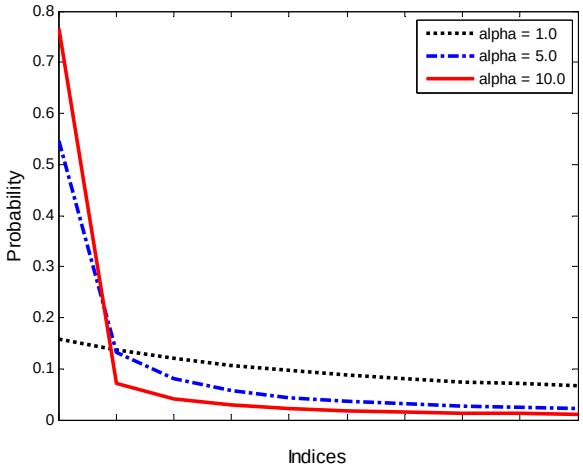


Figure 4.7: Selection probability of IESS with different values of α

are sorted in order of increasing energy values. The selection scheme will determine the probability of selecting each allele of the individuals. It is observed that after IESS, the new formed population consists mainly of the alleles of the individuals with lower energy values. On the other hand, USS selects each allele of the individuals with equal probability.

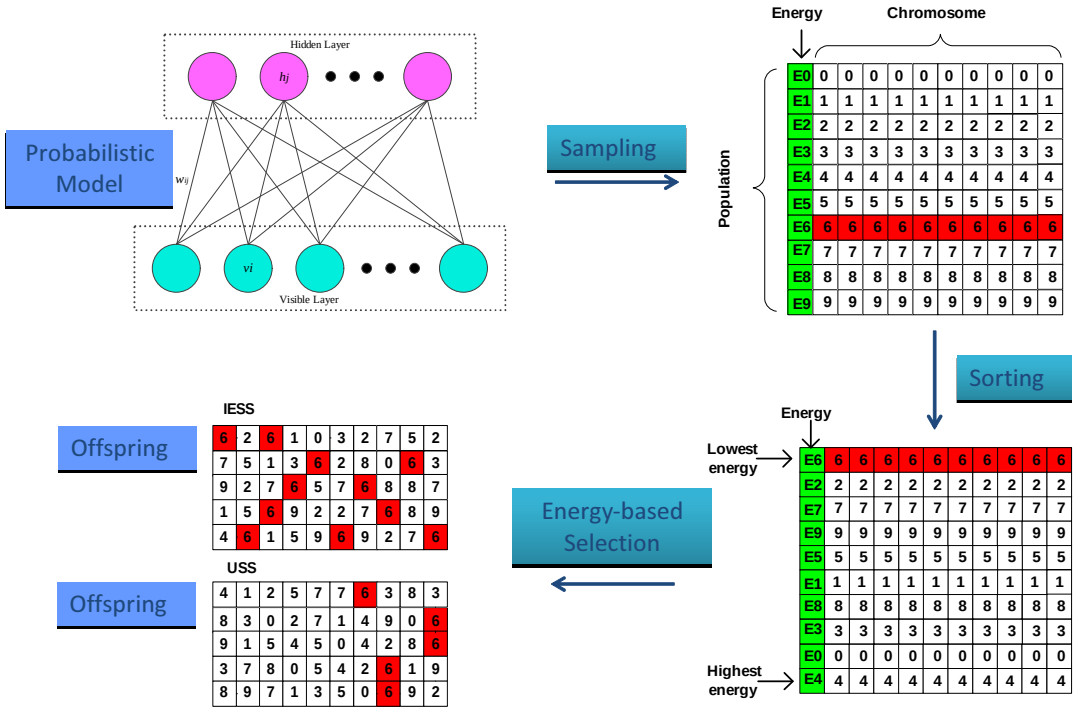


Figure 4.8: Process flow of the energy-based sampling mechanism

4.4 Problem Description and Implementation

This section presents the multi-objective benchmark test problems used to evaluate the effectiveness of the proposed energy-based sampling mechanism. The test problems with artificial linkage dependencies between the decision variables are designed by introducing three types of transformation matrices. The parameter settings and implementation of the simulation runs are also outlined.

4.4.1 Static and Epistatic Test Problems

Static Test Problems

Eight benchmark test instances (F1-F8) with different characteristics have been selected to test the performance of the energy-based sampling mechanism. These problems, F1-F5 [101] and F6-F8 [102], have been used in [38–40, 138–140]. All of them are minimization problems. F1-F5 are variants of ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6, respectively, while F6-F8 are variants of DTLZ1, DTLZ2, and DTLZ3, respectively. The detailed description of the test problems can be referred to Section 2.5.

Epistatic Test Problems

One of the characteristics of EDAs is their ability to capture the interdependencies between the decision variables and use this information to guide the search. In order to show the effectiveness of the energy-based sampling mechanism in dealing with interdependent decision variables, a few transformation matrices are used to introduce artificial linkages between the decision variables of five of the above test problems. Introducing linkage dependencies between the decision variables is one of the ways to increase the difficulty of a problem [97, 141–143]. In this implementation, three types of epistatic problems are designed based on the suggestions provided in [110]. The idea is to introduce a transformation matrix that converts original variables $\mathbf{x}$ into the corresponding variables $\mathbf{y}$. In this way, each of the variables is influenced by some other

variables and their relationships are described by the transformation matrix.

1. **Type-1 problems:** In these problems, the linkages are introduced between the decision variables in $\mathbf{x_I}$ or $\mathbf{x_{II}}$ but no linkage is formed between the decision variables in $\mathbf{x_I}$ and $\mathbf{x_{II}}$, where $\mathbf{x_I}$ is the set of decision variables in cost function I and $\mathbf{x_{II}}$ is the set of decision variables in cost function II. The modified F1 problem with Type-1 linkages is described as follows:

$$f_1(\mathbf{x}) = x_1$$
$$f_2(\mathbf{y}) = g(\mathbf{y})\left(1 - \left(\frac{f_1(\mathbf{x})}{g(\mathbf{y})}\right)^{\frac{1}{2}}\right)$$
$$g(\mathbf{y}) = 1 + \frac{9\left(\sum_{i=2}^{n} y_i\right)}{n-1}$$
$$\mathbf{y} = \mathbf{T}\mathbf{x}$$

   where $\mathbf{T}$ is a $(n-1) \times (n-1)$ matrix with uniformly distributed values between [0, 1]. In this type of problems, there is no linkage between cost functions f_1 and f_2. The explicit linkages are only introduced in the g function, which is a function of f_2. Therefore, all the decision variables, except x_1, are explicitly linked. The same modifications are made to F2 to F5 problems.

2. **Type-2 problems:** In Type-2 problems, linkages are introduced between all the decision variables, thereby making the first cost function correlated to the second cost function. The modified F1 problem with Type-2 linkages is similar to Type-1 problems, except the transformation matrix $\mathbf{T}$ is a $n \times n$ matrix. This modification makes the g function no longer a constant for optimal solution. In order to trace the optimal Pareto front for analysis purposes, we fix the value for T_{i1} as 0.1. Therefore, the optimal g function is obtained as follows:

$$g(\mathbf{x}) = 1 + \frac{9\left(\sum_{i=2}^{n} y_i\right)}{n-1}$$
$$g(\mathbf{x}) = 1 + \frac{9\left(\sum_{i=2}^{n} \mathbf{T}_{i1} x_i\right)}{n-1}$$

$$g(\mathbf{x}) = 1 + \frac{9\left(\sum_{i=2}^{n} \mathbf{T}_{i1} x_1\right)}{n-1}$$

$$g(\mathbf{x}) = 1 + 0.9x_1$$

The same modifications are made to F2 to F5 problems and the optimal fronts are obtained according to the above step.

3. **Type-3 problems:** The above transformation matrices involve only linear transformations. In order to further increase the difficulty of the problem, Type-3 problems involve non-linear mappings between $\mathbf{y}$ and $\mathbf{x}$, as presented below:

$$\begin{bmatrix} y_i \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \mathbf{T}. \begin{bmatrix} \sqrt{x_i} \\ \sqrt{x_2} \\ \vdots \\ \sqrt{x_n} \end{bmatrix}$$

The same modifications are also made to F2 to F5 problems and the Pareto optimal front for F1 problem is given by the g function as follows:

$$g(\mathbf{x}) = 1 + 0.9\sqrt{x_1}$$

4.4.2 Implementation

Three state-of-the-art algorithms, including REDA, NSGA-II, and MOUMDA, are put into comparison against the REDA with the energy-based sampling mechanism proposed in this chapter. REDA/E refers to REDA with IESS while REDA/U is REDA with USS. REDA is the original algorithm that uses a simple sampling technique as presented in previous chapter. NSGA-II is chosen since it is a common benchmark MOEA and it generally achieves good results for most of the test problems. MOUMDA is another MOEDA based on univariate marginal distribution

algorithm (UMDA). The basic architecture of MOUMDA is quite similar to REDA. The main difference is that REDA utilizes multivariate modelling while MOUMDA uses univariate modelling. All the codes were written in C++ and the simulations were performed on an Intel(R) Core(TM)2 Duo CPU, 3.0GHz. The experimental settings are presented in Table 4.1.

Table 4.1: Parameter settings

Parameter	Setting
Population size	100 for all test problems
Stopping criterion	100 generations for static test problems and 300 generations for epistatic test problems
Number of independent runs	10
Number of hidden units in RBM	20 for F1-F5 problems and 5 for F6-F8 problems as suggested in the previous chapter
Number of training epochs in RBM	10 for all test instances as suggested in the previous chapter
Learning rate in REDA	0.1 as suggested in the previous chapter
α in IESS	5.5
Multiplier M in energy-based sampling technique	10
Crossover rate in NSGA-II	0.8
Mutation in NSGA-II	1/(Variable_size $\times$ Variable_bit)

4.5 Simulation Results and Discussions

In this section, comparative studies are conducted to evaluate the performance of the five algorithms on the eight benchmark test instances. The simulation results for the problems with linkage dependencies are also presented. Furthermore, investigations are carried out to analyze the effects of M and α on the effectiveness of the proposed energy-based sampling mechanism. Finally, computational time analysis is conducted. The empirical results are presented using IGD and NR performance indicators. A smaller IGD value implies better performance in terms of closer proximity of the evolvable solutions to the Pareto optimal front and a wider distribution of the evolvable front along the Pareto optimal front. NR measures the ratio of non-dominated solutions among all the solutions generated by all the algorithms, and a higher value indicates that the algorithm produces more non-dominated solutions. Box-plot is used to present the statistical results from 10 independent runs of each of the algorithms. Line curve is used to show the convergence traces. The legend for the convergence trace curve is shown in Figure 4.9. The indices of the algorithms as used in the box-plot are explained in Table 4.2.

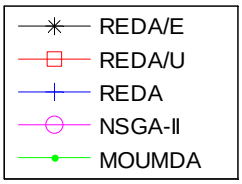


Figure 4.9: Legend for convergence trace curve

Table 4.2: Indices of the algorithms

Index	Algorithm
1	REDA/E
2	REDA/U
3	REDA
4	NSGA-II
5	MOUMDA

4.5.1 Results on Static Test Problems

Figure 4.10(a) shows the convergence traces for the IGD performance indicator from the early stages of evolution up to the 10,000th fitness evaluation for F1 problem with 100 decision variables. It is observed that REDA/E outperforms the other algorithms in both solutions quality (lower value of IGD) and convergence rate (faster convergence). The IGD results at the 10,000th fitness evaluation for the 10 independent simulation runs are presented in Figure 4.10(b). No significant improvement is observed in REDA/U compared to REDA. This may be due to the fact that most of the fitter solutions have lower energy values since lower energy means the solutions are stable under the modelled probability distribution. For REDA/U, all alleles of the individuals have equal chance of being selected. Therefore, the formed offspring consist of an equal number of the alleles of individuals with low and high energy values. This is quite similar to the condition of the original REDA. The implementation of the IESS (REDA/E) means that an allele of a solution with a lower energy value stands a higher chance of being chosen. This results in a better performance. A smaller selection probability is given to the alleles of an individual with a higher energy value, thus increasing the exploratory capability of the algorithm. NSGA-

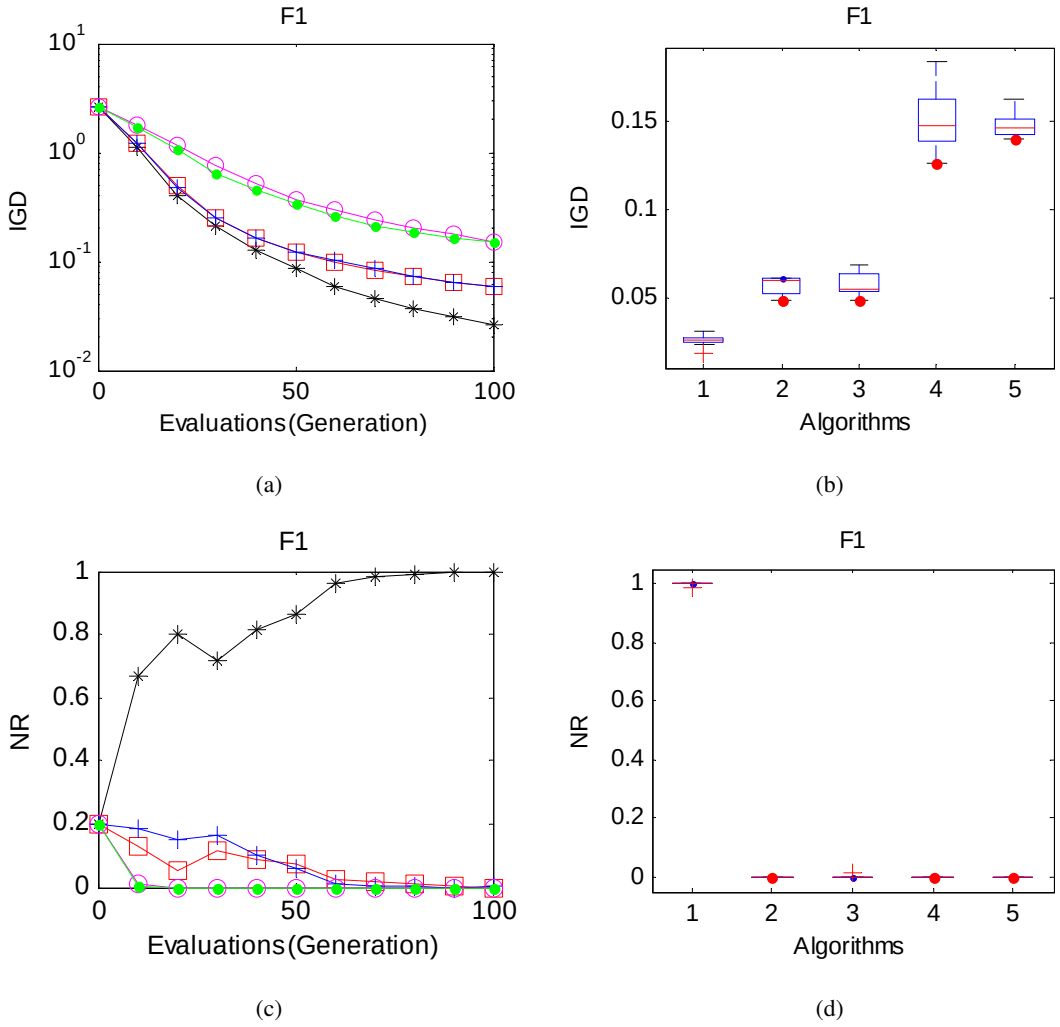


Figure 4.10: Simulation results of various algorithms for F1 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

II and MOUMDA perform worse than all versions of REDA. NSGA-II incorporates stochastic recombination, which requires a larger number of fitness evaluations to evolve better solutions. MOUMDA models the probability distribution, in terms of marginal distribution in each decision variable, without considering any linkage information. It is thus unable to model an overall fit solution set. The change in the NR of each of the algorithms over generations is plotted in Figure 4.10(c). NR values after 10,000 fitness evaluations are presented in box-plot as shown in Figure 4.10(d). It is observed that REDA/E evolves more non-dominated solutions. While the NR for REDA/E increased over generations, the NR for the other algorithms declined. At the end of the simulations, almost all the individuals evolved by REDA/E dominate solutions evolved by

the other four algorithms. REDA and REDA/U maintained a good number of solutions during the early stages of evolution but their solutions were eventually dominated by those of REDA/E. REDA/E is able to maintain a good ratio due to its ability to preserve good solution distribution.

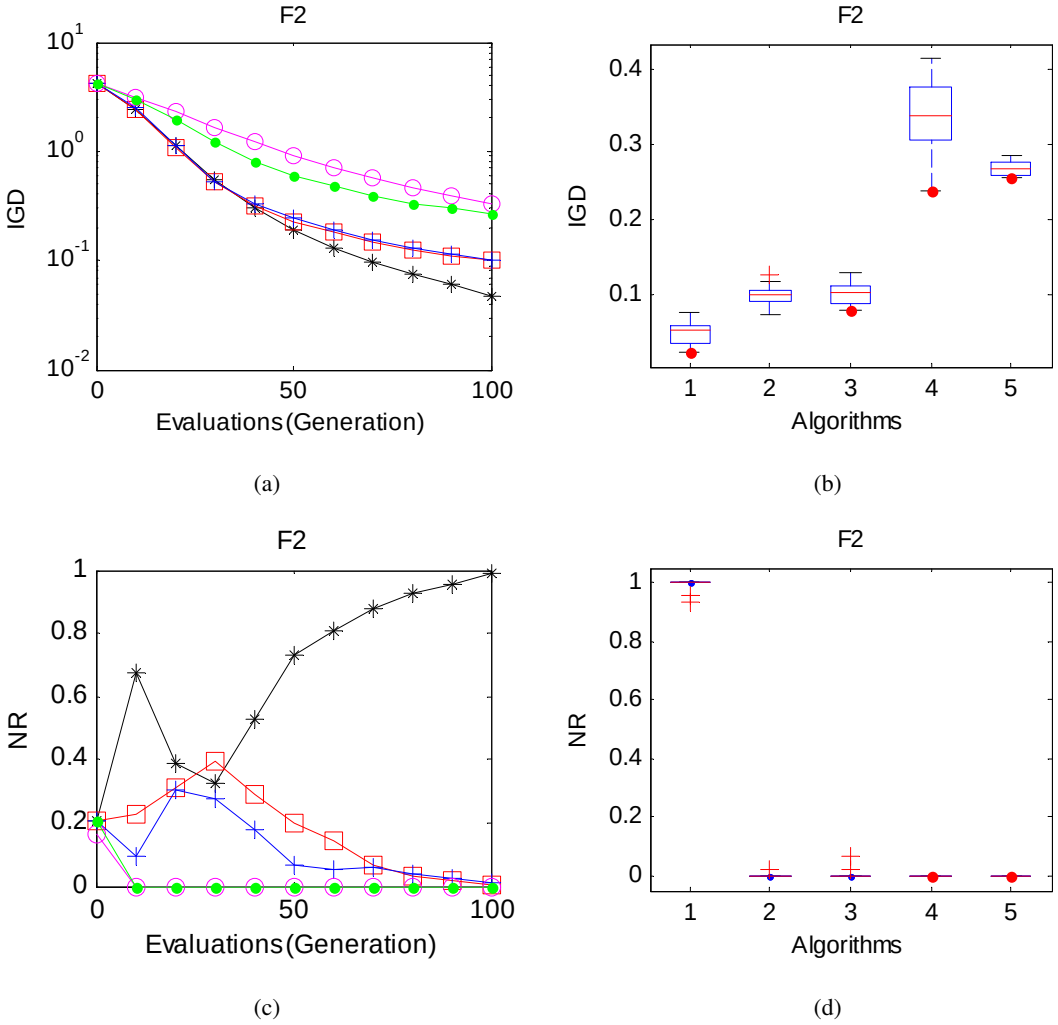


Figure 4.11: Simulation results of various algorithms for F2 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

Figure 4.11 shows the convergence traces and box-plot in terms of IGD and NR performance indicators for F2 problem with 100 decision variables. The results obtained in this problem are quite similar to those obtained in F1 problem. REDA/E showed promising results in terms of convergence rate (Figure 4.11(a)) and IGD performance indicator (Figure 4.11(b)). Again, no significant improvement is seen in REDA/U. For the NR indicator, REDA/E performed well during the early and final stages of evolution, while REDA and REDA/U had comparable

performance during the intermediate stages of evolution. This may be attributed to the fact that the evolvable solutions will distribute in a small phenotypic domain space (due to the characteristic of the problem). In the small region, the distributions modelled by all versions of REDA are quite similar, thus yielding almost equal number of non-dominated solutions.

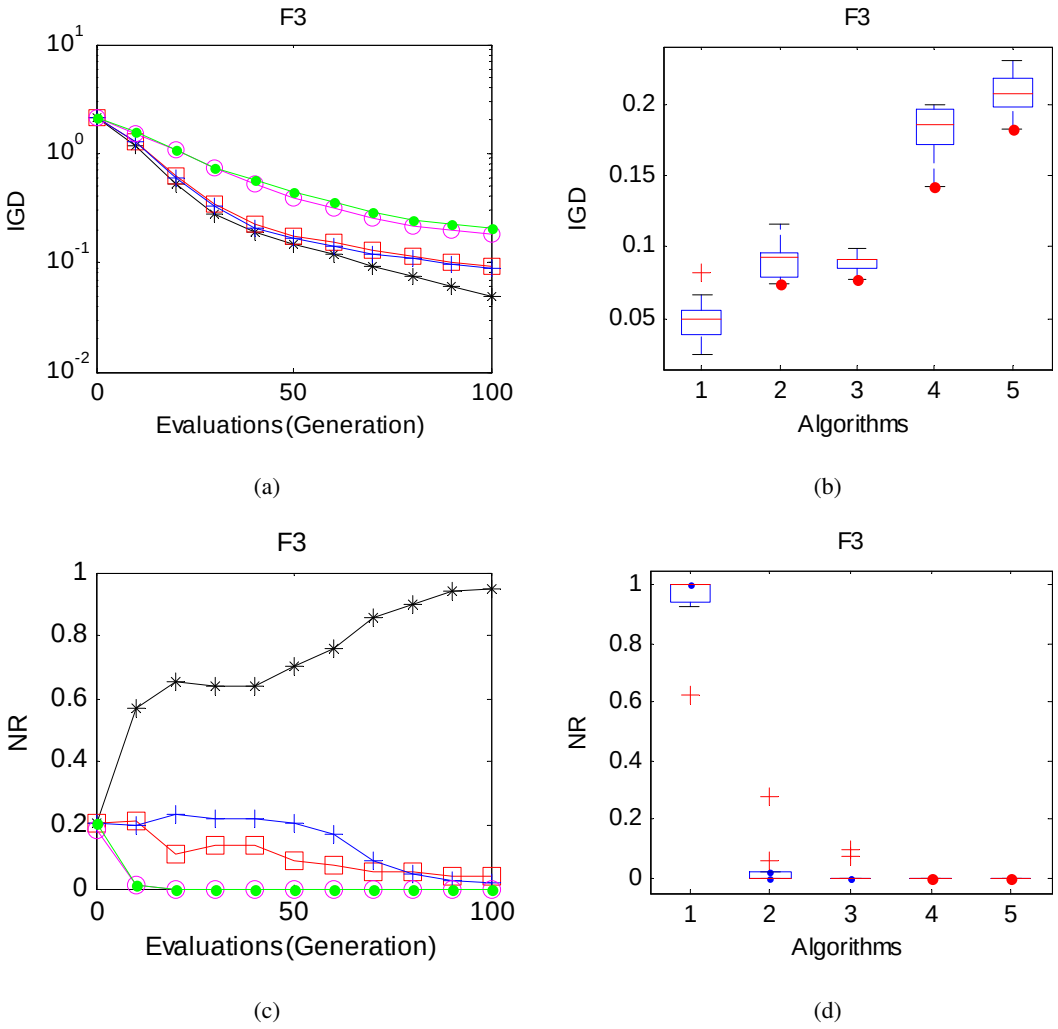


Figure 4.12: Simulation results of various algorithms for F3 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

The results for F3 problem with 100 decision variables are presented in Figure 4.12. This problem has a disconnected and non-convex Pareto optimal front, which may test the ability of the algorithms to find the solutions in different regions as well as maintaining a uniformly distributed solution set. Figure 4.12(a) and 4.12(b), respectively, show that REDA/E outperforms the other algorithm in terms of convergence rate and IGD performance indicator. REDA/U and

REDA yield nearly similar results. Besides, NSGA-II outperforms MOUMDA but the performances of these two algorithms are not as good as that of any of the three versions of REDA. As for NR indicator, a large portion of non-dominated solutions is generated by REDA/E, while a smaller number of non-dominated solutions are generated by REDA/U.

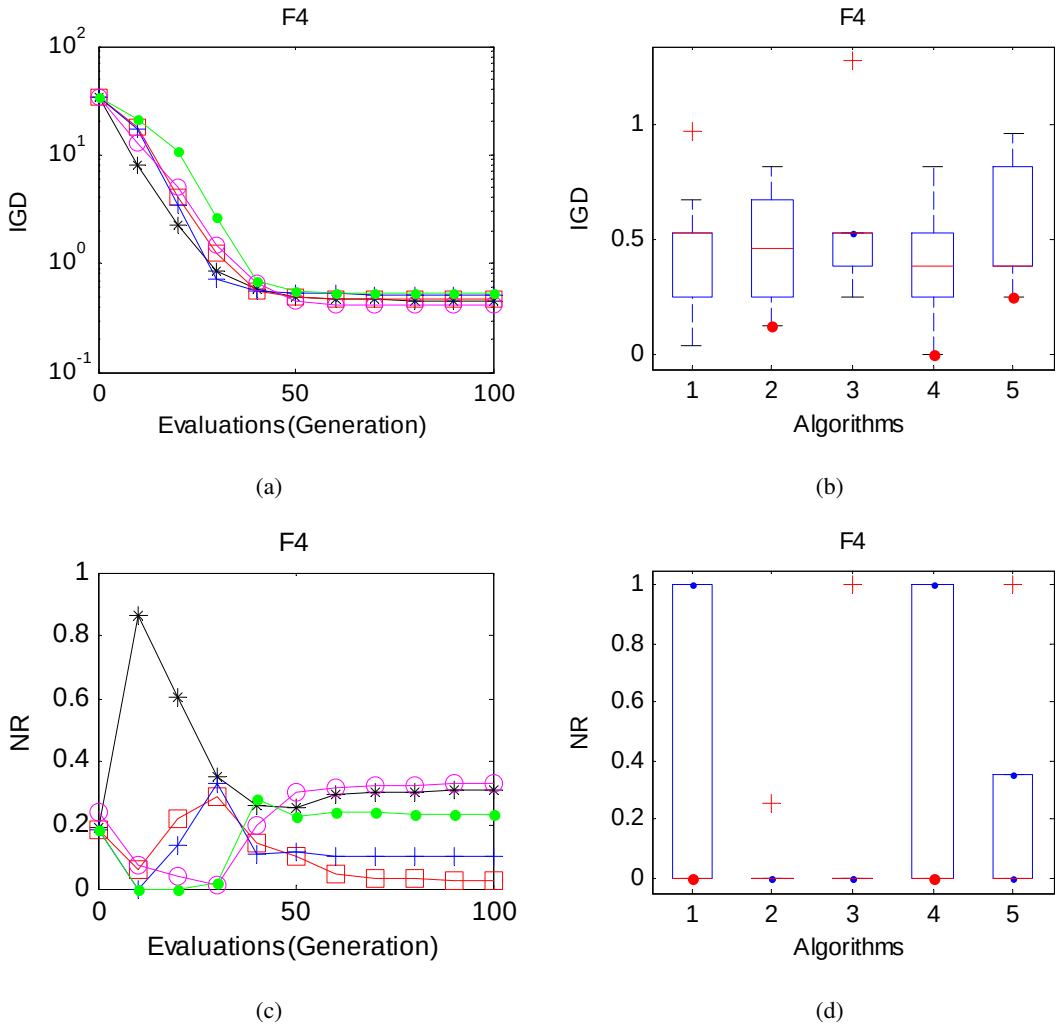


Figure 4.13: Simulation results of various algorithms for F4 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

F4 problem is characterized by its many local optimal fronts. Due to the difficulty of this problem, the number of decision variables is maintained at 10 as suggested in [101]. Figure 4.13 shows the (a) convergence traces and (b) box-plot of the IGD performance indicator after running the algorithms on F4 problem. It can be observed that all the algorithms are unable to produce a global Pareto optimal front. It is likely that the algorithms are trapped in some local

optima. This result suggests that REDA and its variants are unable to deal well with this kind of problem. The sampling mechanism makes use of energy information to enhance the exploration and exploitation ability of the REDA but the information is of little help in dealing with problems with many local optima. Whenever a few solutions are trapped in any local optima, the network will model the distribution of these sub-optimal regions, which will then be treated as promising regions. A hybrid mechanism is probably one of the ways to overcome this limitation. In terms of NR value, REDA/E has a higher ratio during the early stages of evolution but the ratio declines as the search progresses. Even though its overall convergence is not the best, REDA/E is still able to obtain more non-dominated solutions compared to REDA/U and REDA.

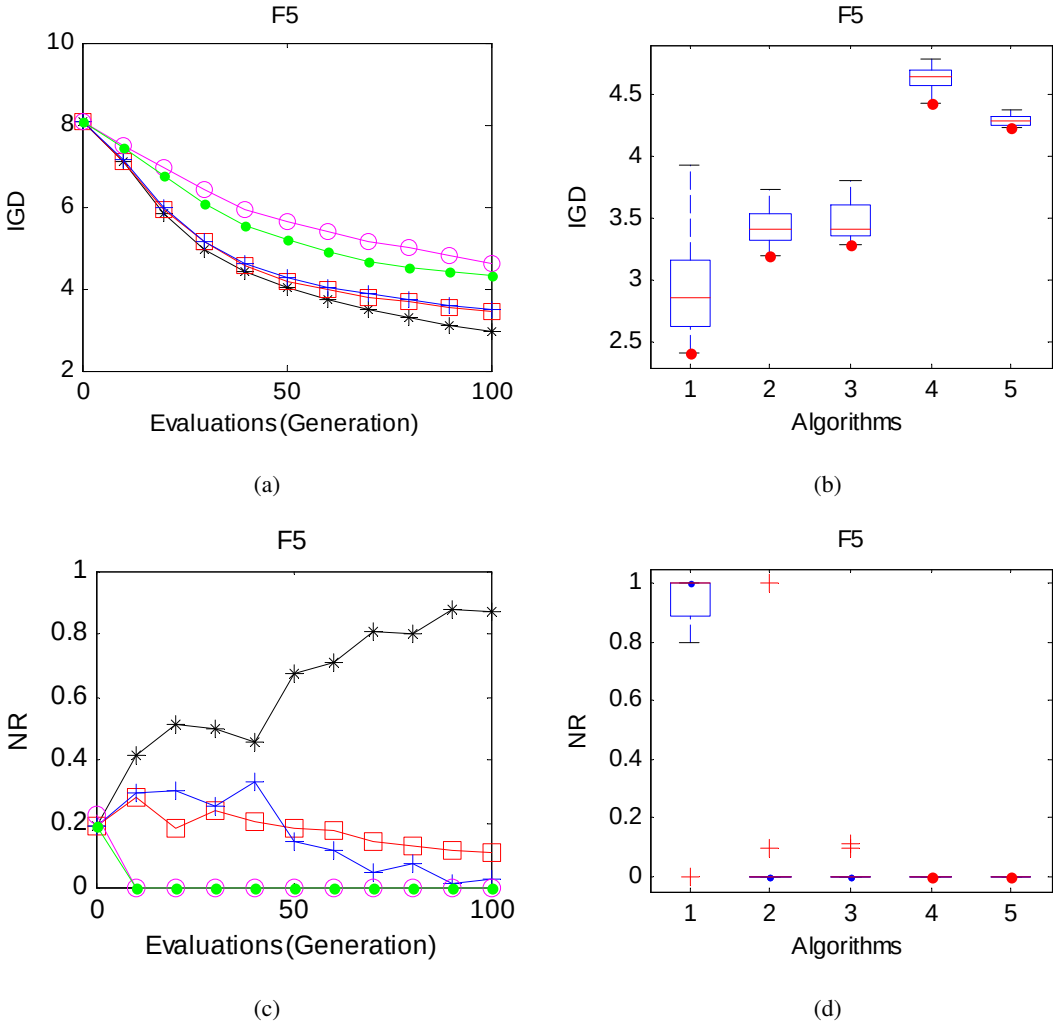


Figure 4.14: Simulation results of various algorithms for F5 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

F5 is another challenging problem that is characterized by a non-uniform and non-convex Pareto optimal front. Figure 4.14 shows the results for F5 problem with 100 decision variables. It is observed that REDA/E gives the best performance in terms of IGD performance indicator (Figure 4.14(b)) and number of non-dominated solutions (Figure 4.14(d)). The improved performance is due to the incorporation of the energy-based sampling mechanism, which allows fitter solutions to be sampled after considering the energy information of the solutions.

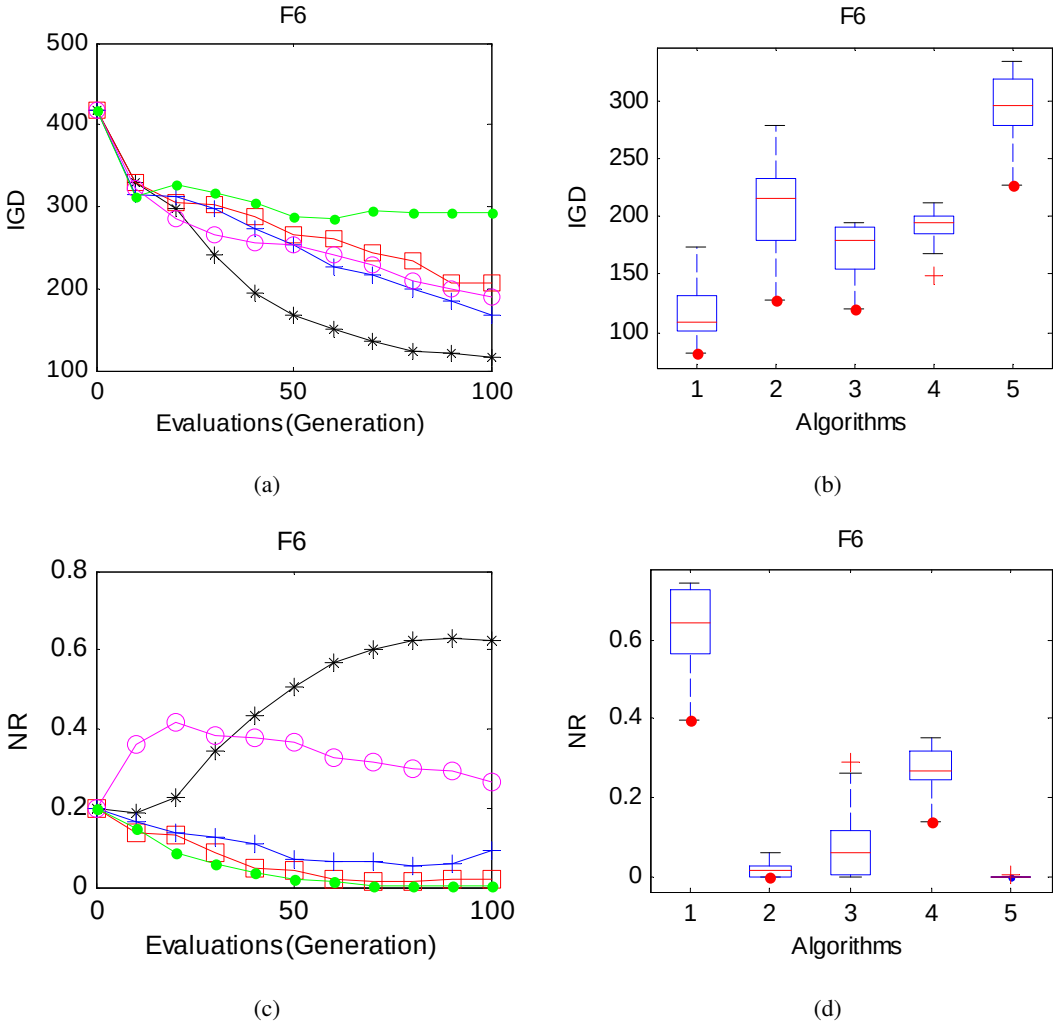


Figure 4.15: Simulation results of various algorithms for F6 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

F6-F8 problems can be scaled up to any number of decision variables and objective functions. Scalable problems will not be dealt with here since they are not within the scope of this chapter. As such, the problems are fixed to three objective functions and 20 decision variables.

F6 problem is well-known to have linear Pareto optimal fronts. Figure 4.15 shows the (a) convergence traces and (b) box-plot of the IGD performance indicator after running the algorithms on F6 problem. It is clear from the results that REDA/E outperforms the other algorithms, in terms of convergence rate, proximity of the evolvable front to the Pareto optimal front, and distribution of solutions on the Pareto optimal front. Besides, NSGA-II and REDA yield similar results. REDA/U, on the other hand, performs slightly worse than REDA. As for the NR performance indicator, NSGA-II seems to have more non-dominated solutions during the early stages of evolution. However, REDA/E is able to eventually evolve a set of fitter solutions that is able to dominate most of the solutions produced by the other algorithms.

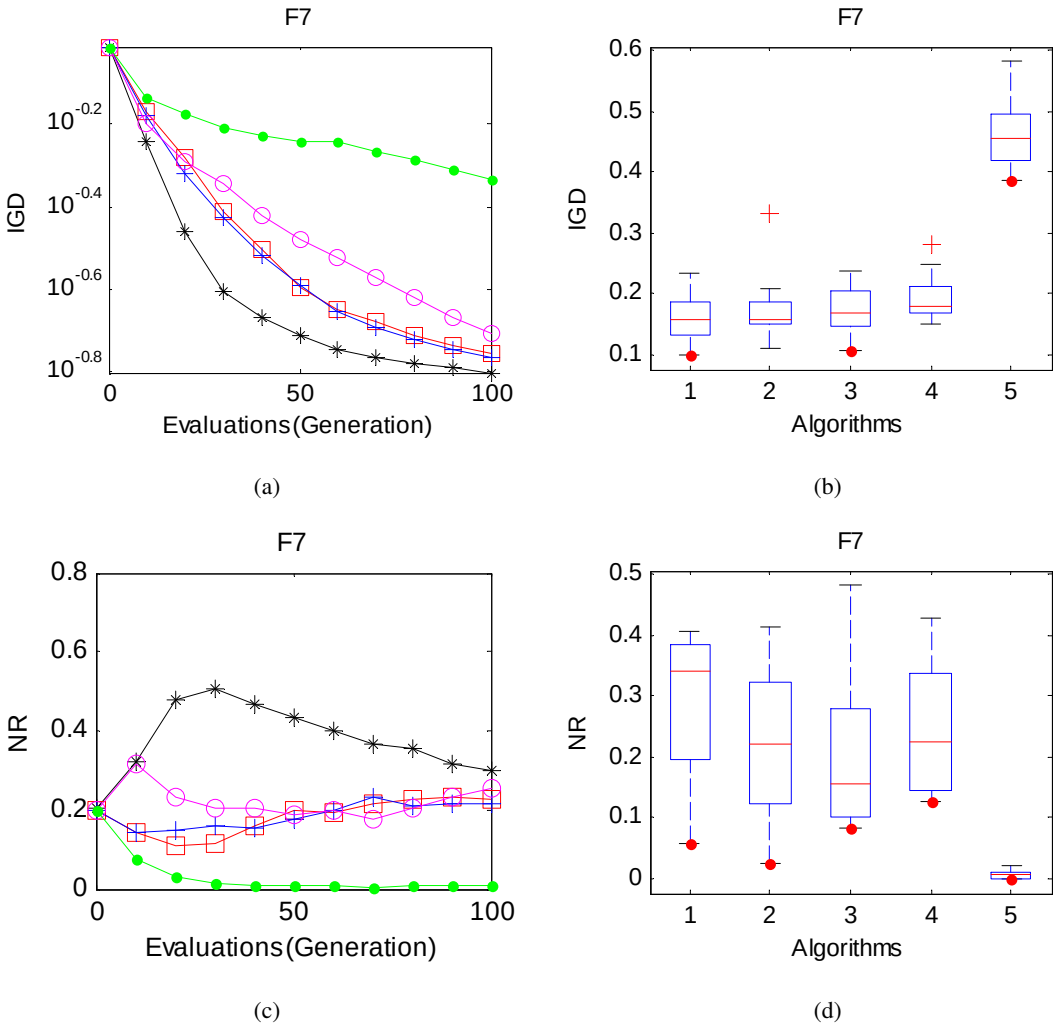


Figure 4.16: Simulation results of various algorithms for F7 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

F7 problem has a spherical Pareto optimal front. The simulation results are presented in Figure 4.16. REDA/E shows promising results in terms of convergence rate and slightly outperforms the other algorithms in terms of both IGD and NR performance indicators. No significant improvement is observed in REDA/U compared to REDA.

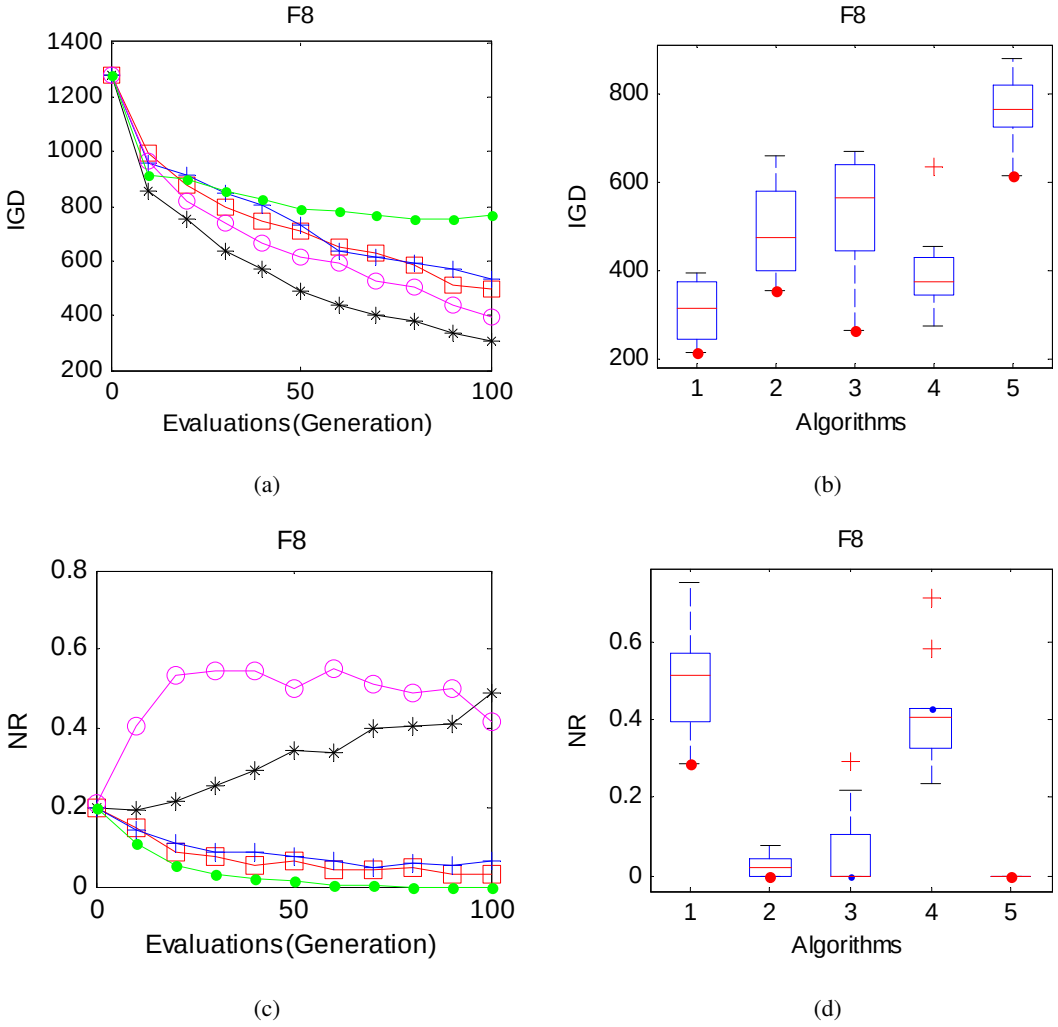


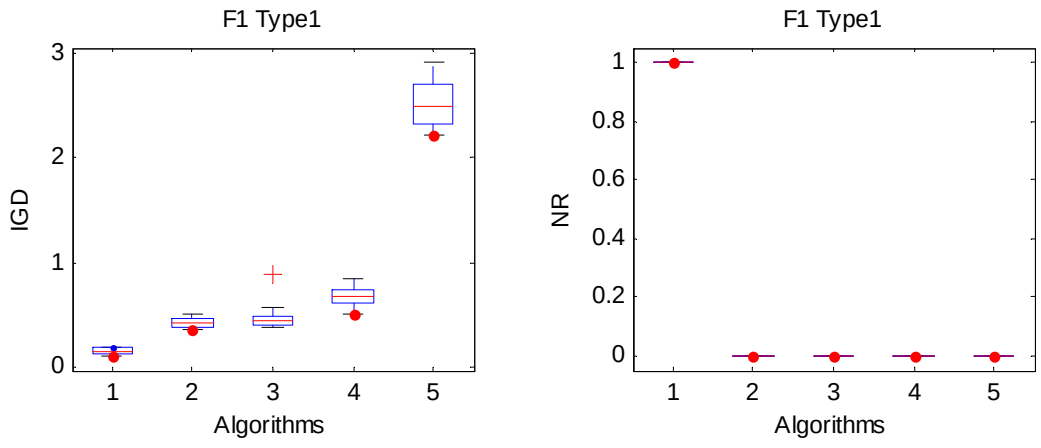
Figure 4.17: Simulation results of various algorithms for F8 problem (a) Convergence traces for IGD measurement (b) IGD values after 10,000 fitness evaluations (c) Convergence traces for NR measurement (d) NR values after 10,000 fitness evaluations

F8 problem has a spherical Pareto front and consists of many local optima. The results obtained by the various algorithms for F8 problem are shown in Figure 4.17. From the results, it can be seen that the performance of REDA is significantly improved after incorporating the energy-based sampling mechanism. Specifically, the performance improvement brought about by the IESS is more significant. The results measured in terms of NR are illustrated in Figures

4.17(c) and 4.17(d). From the figures, it can be observed that REDA/E is able to gradually evolve non-dominated solutions and eventually perform better than NSGA-II, which has been generating most of the non-dominated solutions throughout the runs. From this observation, it can be concluded that the energy-based sampling mechanism with IESS succeeded in improving the performance of REDA while the sampling technique with USS did not result in significant improvement. This observation suggests that greater selection pressure should be assigned to the alleles of solutions with lower energy values, while smaller selection pressure should be assigned to the alleles of individuals with higher energy values.

4.5.2 Results on Epistatic Test Problems

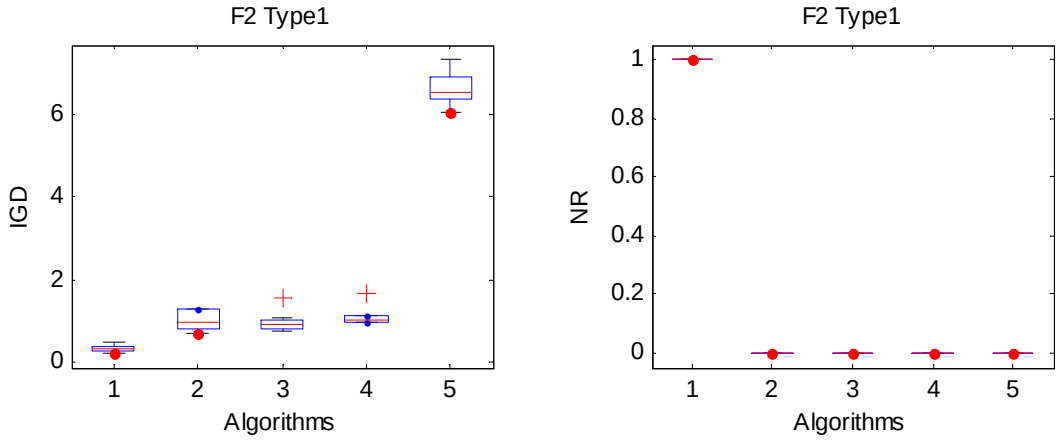
In order to show the ability of the energy-based sampling mechanism in solving epistatic test problems, a number of transformation matrices are used to modify the static test problems. Only F1-F5 problems are used in this study. From preliminary studies, it is observed that the convergence rates of the algorithms decrease when solving the epistatic problems. Therefore, the algorithms are allowed to run longer by setting the stopping criterion to be 30,000 fitness evaluations instead of 10,000 fitness evaluations.



(a) (b)

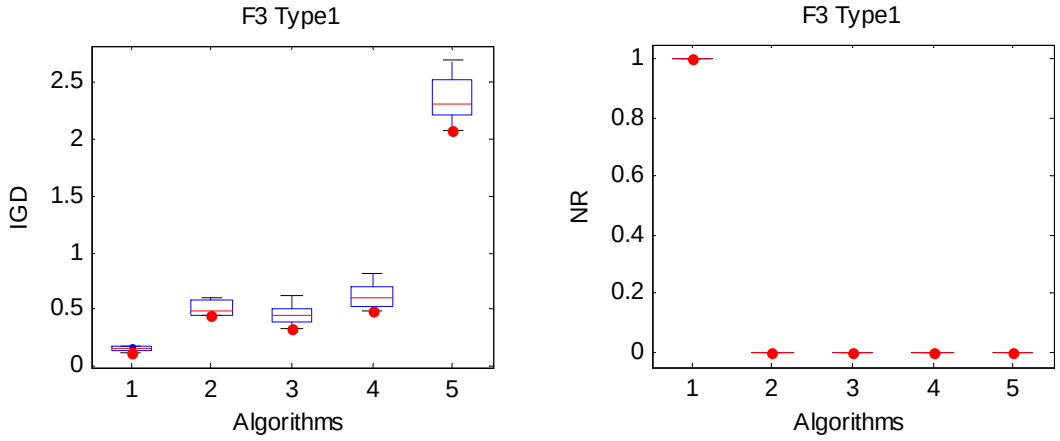
Figure 4.18: (a) IGD and (b) NR performance metrics obtained for F1 Type-1 problem after 30,000 fitness evaluations

Figure 4.18 shows the (a) IGD and (b) NR results achieved by the algorithms after 30,000



(a) (b)

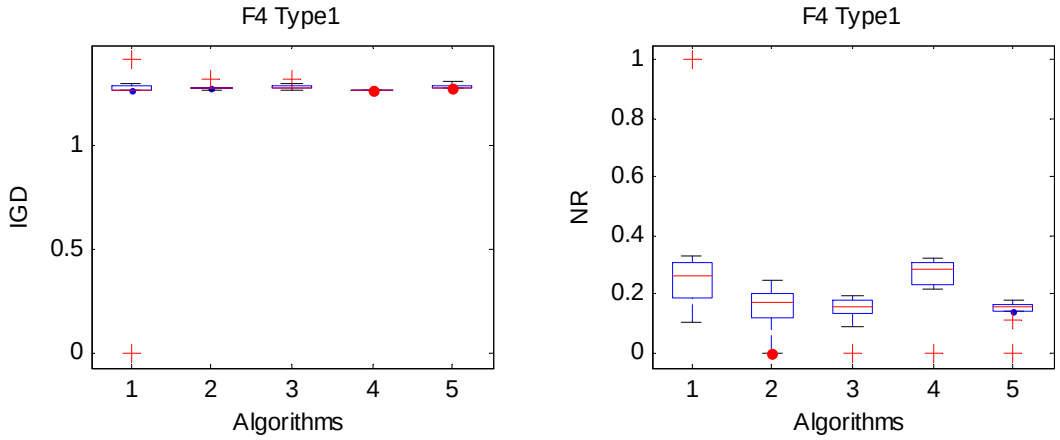
Figure 4.19: (a) IGD and (b) NR performance metrics obtained for F2 Type-1 problem after 30,000 fitness evaluations



(a) (b)

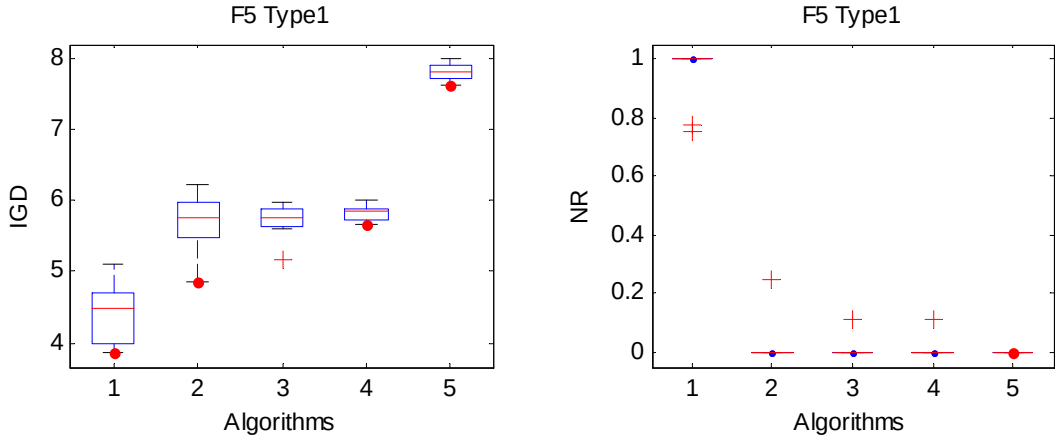
Figure 4.20: (a) IGD and (b) NR performance metrics obtained for F3 Type-1 problem after 30,000 fitness evaluations

fitness evaluations for F1 Type-1 problem. It is observed that all variants of REDA performed better than NSGA-II and MOUMDA. REDA models the probability distribution of the solutions by applying the energy information to detect the dependencies and is able to perform well on epistatic problems. The incorporation of IESS in energy-based sampling mechanism enhances the exploration and exploitation capability of the algorithm and REDA/E outperforms the other REDA variants in terms of IGD and NR performance metrics. Similar results are observed in F2-F5 Type-1 problems, with the exception of F4 Type-1 problem, as presented in Figures 4.19-



(a) (b)

Figure 4.21: (a) IGD and (b) NR performance metrics obtained for F4 Type-1 problem after 30,000 fitness evaluations



(a) (b)

Figure 4.22: (a) IGD and (b) NR performance metrics obtained for F5 Type-1 problem after 30,000 fitness evaluations

4.22. For F4 Type-1 problem, REDA/E is able to generate better solutions than the other REDA variants and its performance is comparable to NSGA-II. However, all the algorithms are trapped in local optima. It can be observed in Figure 4.21 that one of REDA/E's simulation runs produced significantly better results. Therefore, it can be concluded that REDA/E is sensitive to different initializations for this particular problem. For ease of visualization, the average IGD and NR results of the various algorithms over the 10 simulation runs for Type-2 and Type-3 problems are tabulated in Table 4.3. The best results are highlighted in bold. REDA/E gives the best

Table 4.3: Results obtained by five algorithms for Type-2 and Type-3 problems

		Type-2		Type-3	
Problem	Algorithm	IGD	NR	IGD	NR
F1	REDA/E	**0.2138**±0.0372	**1.0000**±0.0000	**2.3730**±0.5129	**0.8583**±0.3144
	REDA/U	0.5508±0.1394	0.0000±0.0000	3.1722±0.7394	0.1417±0.3144
	REDA	0.5515±0.1093	0.0000±0.0000	4.1947±0.9252	0.0000±0.0000
	NSGA-II	0.7558±0.1056	0.0000±0.0000	10.7000±1.1603	0.0000±0.0000
	MOUMDA	2.7427±0.2067	0.0000±0.0000	19.6738±0.7784	0.0000±0.0000
F2	REDA/E	**0.1582**±0.0486	**0.8594**±0.0631	**0.2948**±0.0113	**0.7375**±0.1755
	REDA/U	0.3397±0.0416	0.0298±0.0321	0.3178±0.0108	0.0770±0.0908
	REDA	0.3297±0.0318	0.0256±0.0459	0.3106±0.0108	0.1406±0.1252
	NSGA-II	0.3186±0.0259	0.0728±0.0799	0.3376±0.0029	0.0279±0.0093
	MOUMDA	0.4515±0.0082	0.0124±0.0140	0.3502±0.0055	0.0170±0.0120
F3	REDA/E	**0.2602**±0.0907	**0.9584**±0.0928	**2.4056**±0.3309	**0.9308**±0.1315
	REDA/U	0.5773±0.1342	0.0273±0.0862	3.9842±0.9489	0.0525±0.1283
	REDA	0.5913±0.1239	0.0143±0.0452	4.5530±0.9709	0.0167±0.0527
	NSGA-II	0.6877±0.1091	0.0000±0.0000	11.197±1.2721	0.0000±0.0000
	MOUMDA	2.7673±0.2987	0.0000±0.0000	20.152±1.2352	0.0000±0.0000
F4	REDA/E	1.4058±0.4639	**0.3309**±0.3842	1.5299±0.0919	0.1200±0.0259
	REDA/U	**1.3751**±0.4091	0.2214±0.3106	1.4805±0.0551	0.1562±0.0300
	REDA	1.6079±0.1345	0.1093±0.1325	1.5171±0.0963	0.1456±0.0361
	NSGA-II	1.5615±0.1201	0.1289±0.1506	**1.3626**±0.0789	**0.4085**±0.0344
	MOUMDA	1.4639±0.3522	0.2095±0.3090	1.4675±0.0737	0.1698±0.0229
F5	REDA/E	**1.6526**±0.2638	**1.0000**±0.0000	**3.6949**±0.4130	**0.9496**±0.0684
	REDA/U	2.9464±0.3075	0.0000±0.0000	4.5404±0.4427	0.0504±0.0684
	REDA	2.8769±0.1664	0.0000±0.0000	4.8418±0.3496	0.0000±0.0000
	NSGA-II	3.2577±0.3095	0.0000±0.0000	6.8923±0.1827	0.0000±0.0000
	MOUMDA	5.0655±0.1704	0.0000±0.0000	8.1999±0.1338	0.0000±0.0000

performance in all the test problems, with the exception of F4 Type-2 and Type-3 problems. The performance of all the algorithms deteriorated in Type-3 problems where the IGD values were greater than those for the corresponding Type-1 and Type-2 problems. This is attributed to the introduction of a non-linear transformation matrix as in Type-3 problems whereas a linear transformation matrix is applied in Type-1 and Type-2 problems.

4.5.3 Effects of Decay Factor of Inverse Exponential Selection Scheme on Optimization Performance

α determines the decay factor of an exponential function ($e^{\alpha x}$). If x is kept constant, α will be the only parameter that determines the probability of selection as shown in Figure 4.7. In this section, the effect of different values of α on optimization performance is examined. Figure 4.23 shows the convergence traces of REDA/E using different values of α to solve (a) F1 and (b) F6 problems. It is observed that an α value that is too low (1.0) or too high (9.0) would not yield good results, while an α value ranging from 3.0 to 7.0 gives acceptable performance. Recall that a lower value of α will give a more even probability distribution function used for selecting the

alleles of solutions, while a higher value of α will give a higher chance of selecting the alleles of solutions with lower energy values. A set of offspring will contain more unfit individuals if all the alleles of solutions are given almost equal chances of being selected. On the other hand, if the selection scheme only selects the alleles of individuals with lower energy, less exploration of the search space is performed by the algorithm. A set of offspring with too many alleles of solutions having low or high energies is not an ideal case in an evolutionary process since exploration and exploitation of the search space must be balanced. Therefore, an α value ranging from 3.0 to 7.0 is the ideal setting for implementation.

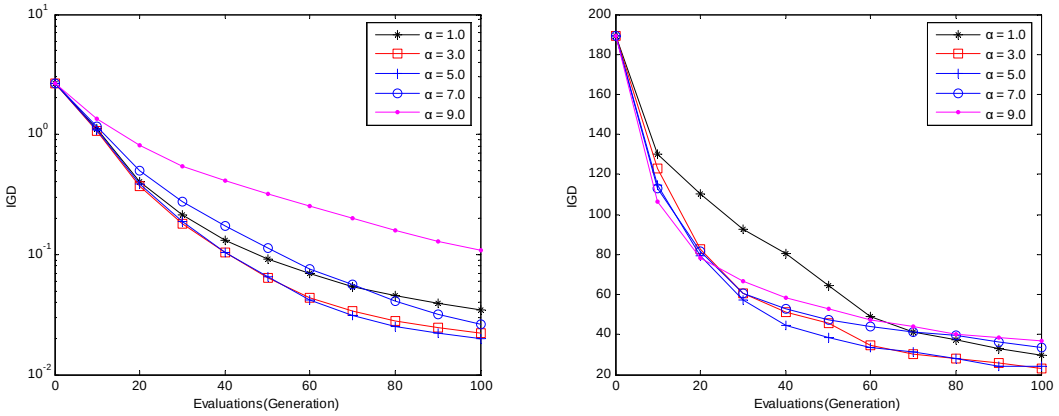
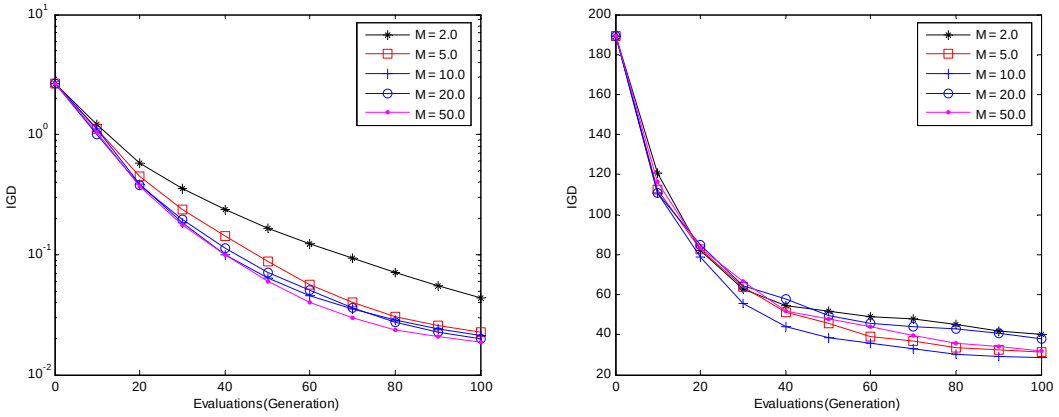


Figure 4.23: Convergence traces of REDA/E for solving (a) F1 and (b) F6 problems under different settings of α

4.5.4 Effects of Multiplier of Energy-based Sampling Mechanism on Optimization Performance

The fundamental idea behind the proposed energy-based sampling mechanism is that sampling the population for an infinitely large number of solutions may increase the chance of producing fitter candidate solutions. However, such a sampling scheme is impractical. Thus, a multiplier M is used to determine the number of sampled individuals. It may be argued that M can affect the final performance of the algorithm. Thus, this section carries out an investigation to examine the effect of M on optimization performance and then suggest a possible range of values for M.

Figure 4.24 shows the convergence traces of REDA/E using different values of M to solve (a) F1 and (b) F6 problems. It is observed that a larger value of M (20 to 50) gives better performance in solving F1 but not F6 problems. A smaller value of M (2) leads to poorer performance while an M value around 5 to 10 generally gives good results for both the test instances. Therefore, it is possible to conclude that large values of M are unnecessary and an M value of around 5 to 10 is enough to evolve a good Pareto front. In fact, the M and α settings are co-related. The purpose of these two parameters in the energy-based sampling mechanism is to assign a higher chance of selecting the alleles of solutions with lower energy values while still allowing some chance for the alleles of solutions with higher energy values to be selected.



(a) (b)

Figure 4.24: Convergence traces of REDA/E for solving (a) F1 and (b) F6 problems under different settings of M

4.5.5 Computational Time Analysis

It is clear that the energy-based sampling mechanism will incur additional computational time due to the sampling of $N \times M$ candidate solutions rather than N solutions, as well as the re-computation of the hidden states of the RBM. A computational time analysis is carried out to determine how costly the proposed sampling technique is. Table 4.4 presents the computational times required by one generation of REDA/E using different settings of M in solving F1 and F6

problems. From the table, the computational time is increased approximately three times when M is 50. However, only a slight increment in computational time is incurred for smaller values of M (2-10). Since an M value of 5-10 is able to give good results, the additional computational time incurred can be considered as insignificant.

Table 4.4: Computational time (in second) used by REDA/E under different settings of M

Problems	Multiplier M					
	1	2	5	10	20	50
F1	2.0772	2.1510	2.3882	2.7551	3.4463	5.5789
F6	0.1670	0.1787	0.1958	0.2365	0.2981	0.5007

4.6 Summary

In this study, an energy-based sampling mechanism for REDA has been proposed. The sampling approach takes advantage of the energy information of solutions. Energy is used to determine a set of offspring to undergo the evolution process in the next generation. The set of selected offspring is a combination of the alleles of solutions with low and high energy values. Two variants of the proposed selection scheme have been presented. Uniform selection scheme (USS) gives an equal chance of selecting any alleles of the solutions. On the other hand, inverse exponential selection scheme (IESS) assigns a larger probability of selecting the alleles of solutions with lower energy values and a smaller probability of selecting the alleles of individuals with higher energy values. The experimental results showed that the energy-based sampling mechanism with IESS improves the performance of REDA in terms of IGD, NR, and convergence rate but at the expense of a longer computational time. Further analyses have also been performed to examine the parameter settings of the energy-based sampling mechanism. Even though the energy-based sampling mechanism has improved the performance of the REDA, the experimental results showed that the algorithm is still unable to converge to the global Pareto optimal front for problems with many local optima. Further investigations are required to overcome this limitation of the algorithm.

Chapter 5

A Hybrid REDA in Noisy Environments

Many real-world optimization problems are subjected to uncertainties that may be characterized by the presence of noise in the objective functions. This chapter studies the potential of REDA that handles multi-objective optimization problems in noisy environments. The focus will be placed to highlight the detrimental effect of noise, proposed noise handling feature via a likelihood correction mechanism, and hybridization with a particle swarm optimization algorithm. The effectiveness of the proposed algorithm is examined via eight benchmark instances with different characteristics and shapes of the Pareto optimal front. The scalability, hybridization, and computational time are rigorously studied.

5.1 Introduction

Though MOEAs have gained a satisfactory performance for certain static optimization problems, the implementation of MOEAs in a noisy environment still requires major study. In noisy environments, the presence of noise in the cost functions may affect the ability of the algorithms to drive the search process towards optimality. To understand the detrimental effect of noise in

evolutionary optimization processes, Bayer [144] carried out an investigation and found that the presence of noise may reduce the convergence rate, resulting in suboptimal solutions. In another study by Goh and Tan [46], it was reported that the low level of noise helps an MOEA to produce better solutions for some problems, but a higher noise level may degenerate the optimization process into a random search. In [145], the authors concluded that re-sampling can reduce the effect of noise for a small population, but may not be as helpful for a larger population. In order to reduce the effect of noise in evolutionary processes, a number of noise handling approaches such as probability dominance [146, 147], fitness inheritance [148], and Dempster-Shafer framework of dominance [149] have been proposed.

While the implementation of standard MOEAs in a noisy environment has gained some attention from researchers, the adoption of MOEDAs in this particular case remains unexplored. As such, this thesis attempts to study the performance of MOEDAs under the influence of noise. The advantage MOEDAs have over standard MOEAs in handling noisy information comes from the construction of noise handling features in the probabilistic model [150]. Hong *et al.* [151] showed that when a smoothing filter is incorporated to an EDA, it is more suited to tackle noisy optimization problems than a genetic algorithm (GA) in single-objective optimization. More specifically, the authors argued that univariate marginal distribution algorithm (UMDA), one of the simplest EDAs, is able to converge to global optimality in the One-max problem under the influence of noise.

In this study, REDA is implemented to tackle multi-objective optimization problems in noisy environments. However, REDA uses global information only in guiding the search, which may not explore the search space thoroughly. In addition, REDA may be trapped at local optima as described in previous chapter. To overcome this limitation, REDA is hybridized with a particle swarm optimization (PSO) [139, 152] algorithm. The PSO is another stochastic computing algorithm inspired by swarm intelligence of insects in forming groups and movements such as birds, fishes, etc. PSO is chosen because of its ease of implementation and lessened sensitivity

to parameter settings. Besides, PSO has already been successfully applied to many real-world optimization problems [153,154] due to its satisfactory search performance and fast convergence rate. The hybridization is expected to improve the performance since the particles may move out of the region modelled by the probabilistic model, providing extra solutions which are unexplored by the model. On the other hand, the presence of noise may affect the selection operator, resulting in a wrong decision. This may directly affect the plausibility of the probabilistic model and cause premature convergence or make the algorithm trapped at local optima. Therefore, a likelihood correction feature is proposed to tune the marginal probability in each decision variable. The likelihood correction takes advantage of the probability error. The probability error is calculated from the model suggested in [146] to determine the probability of making a wrong decision during the selection process due to the presence of noise.

The rest of this chapter is as follows. Section 5.2 describes the problem formulation of noisy MOPs and the existing work on the noise handling features. Section 5.3 presents the framework of the proposed algorithm. Test problems and parameter settings for experiments are outlined in Section 5.4. Section 5.5 examines the performance of the algorithms under the influences of noise, scalability, hybridization, and computational time. The summary is presented in Section 5.6.

5.2 Background Information

5.2.1 Problem Formulation

In this thesis, noise is introduced into the fitness functions in the objective space as Gaussian noise, $N(0, \sigma^2)$, with zero mean and different noise levels represented by variance (σ^2). Mathematically, the noisy fitness function is modelled as:

$$F(x) = f(x) + N(0, \sigma^2) \tag{5.1}$$

5.2.2 Existing Studies

Over the past few years, several noise handling techniques have been proposed in the area of utilizing MOEAs to reduce the detrimental effect of noise. Hughes [146] proposed the multi-objective probabilistic selection evolutionary algorithm (MOPSEA) where the ranking process is reformulated by incorporating the probability dominance among the solutions. The modified ranking process showed promising results in reducing the disturbance of noise. In another independent study by Teich [147], a concept of probabilistic dominance to deal with uncertain objective values constrained within certain intervals, was introduced. This concept was implemented in strength Pareto evolutionary algorithm (SPEA) and called estimate SPEA (ESPEA).

Büche *et al.* [155] proposed a noise-tolerant SPEA (NTSPEA). The study focused on improving the robustness of the solutions when the objective functions are subjected to noise and outliers. Three features were incorporated, including domination dependent lifetime, re-evaluation of solutions, and update of the archive population. A lifetime, which is dependent on the fraction of archived solutions it dominates, is assigned to each of the non-dominated individuals. The solutions in the archive will be removed once the lifetime has expired.

Goh *et al.* [46] carried out an extensive investigation to understand the effect of noise in evolutionary multi-objective optimization. They found that the decision error ratio for selection, ranking, and archiving is lower in the early stages of an evolution, the number of non-dominated solutions found in an archive is reduced when the noise level increases and the average of the population distribution remains invariant in the decision space. Based on these observations, three noise handling features were proposed. Experiential learning directed perturbation (ELDP) performs the mutation paradigm by deciding which gene will undergo mutation. This decision is based on the posterior knowledge of the favourable movements in the search space. Gene adaptation selection strategy (GASS) adopts the feature to prevent search processes from premature convergence or degeneration into random searches. Finally, a possibilistic archiving methodology updates the archive by considering the probability that the domination by an individual is

wrong.

Recently, Bui *et al.* [156] adapted local models in dealing with noisy multi-objective optimization problems. In the proposed algorithm, the search space is divided into several non-overlapping hyper-spheres in order to limit the search within the spheres. Noise filtering was carried out in each hyper-sphere and the future movement of the sphere was adjusted according to the direction of improvement. This implementation reduces the effect of noise based on the movement of the spheres. Other approaches that have been proposed to reduce the effect of noise are fitness inheritance [148], indicator model [157], stochastic dominance and significant dominance [158], Dempster-Shafer framework of dominance [149], and confidence-based operators [159, 160], among others.

5.3 Proposed REDA for Solving Noisy MOPs

REDA surpasses the standard MOEAs in handling noisy information by constructing a noise handling feature in the built probabilistic model. In order to show this advantage, likelihood correction is proposed in order to tune the error in the constructed probabilistic model. Furthermore, REDA is hybridized with PSO in order to improve its search ability. In this section, the framework of the proposed algorithm and other relevant features will be discussed.

5.3.1 Algorithmic Framework

The proposed algorithm incorporates PSO and likelihood correction in REDA and is called PLREDA. The pseudo-code of the PLREDA is presented in Figure 5.1 and the algorithm works as follows. Firstly, at generation $g = 0$, N initial individuals are randomly generated to form an initial population, $Pop(g = 0)$. All the solutions in $Pop(g)$ are evaluated to calculate their objective values. Based on the objective values, Pareto ranking and crowding distance [32] are performed over the solutions. Next, binary tournament selection [161, 162] is carried out. In each selection process, the probability error in selecting a particular individual is computed ac-

cording to equation (5.7). Based on this information, the population is clustered into several groups according to equation (5.8). The population is then rendered into RBM. The RBM is subsequently trained by using the contrastive divergence (CD) [119] training method to obtain the corresponding weights and biases. The probabilistic model using likelihood correction is then built according to equation (5.9). From the constructed probabilistic model, N new offspring are sampled according to equation (3.16) and the solutions are stored in archive $A1$. The solutions in $Pop(g)$ and $A1$ are combined to form a pool of $2N$ solutions. Subsequently, N solutions with the lowest Pareto rank and highest crowding distance are selected from the pool to form the new $Pop(g)$. Next, PSO is performed to update the solutions in $Pop(g)$ and the generated N new solutions are stored in archive $A2$. The solutions in $Pop(g)$ and $A2$ are then combined to form a pool of $2N$ solutions. N solutions with the lowest Pareto rank and highest crowding distance are selected from the pool to form the next population $Pop(g+1)$. The same process is continued until the terminating criterion is reached.

5.3.2 Particle Swarm Optimization (PSO)

REDA has its limitations in exploring the search space and may be trapped at any local optima as investigated in Chapters 3 and 4. In order to overcome these limitations, hybridization is carried out. In the implementation stage, REDA is hybridized with a particle swarm optimization (PSO) algorithm. This hybridization is expected to improve the performance since the particles may now move out of the regions modelled by REDA, and thus provide extra solutions that REDA alone was not able to tap on.

PSO is another stochastic computing algorithm that is inspired by the swarm intelligence of several animals presenting collective behaviour, such as birds, fishes, etc. PSO has gained much interest from research community, in solving real-world optimization problems, due to its ease in implementation and great search performance. PSO is particularly efficient for problems presented in real values. The conversion of PSO from real-number representation to binary-

Begin

1. **Initialization:** At generation $g = 0$, randomly generate N solutions as the initial population, $Pop(g)$; generate empty archives $A1$ and $A2$
2. **Evaluation:** Evaluate all solutions in $Pop(g)$

Do While ("Stopping Criterion is not satisfied")

3. **Fitness assignment:** Perform Pareto ranking and crowding distance to the population $Pop(g)$
4. **Selection:** Select N parent solutions using binary tournament selection
   Probability Dominance: For every solution that win the tournament, calculate its probability of making wrong selection
5. **Clustering:** Group the selected parent solutions into Gp clusters using equation (5.8)
6. **Training:** Train the RBM using CD training mechanism to obtain the weights and biases
7. **Modeling Using Likelihood Correction:** Compute the probability distribution of the solutions, $p_g(\boldsymbol{x})$, using equation (5.9)
8. **Sampling:** Sample N offspring from $p_g(\boldsymbol{x})$ using simple sampling technique. Store the offspring in $A1$
9. **Evaluation:** Evaluate all offspring in $A1$
10. **Archiving:** Combine parent and offspring $Pop(g) \cup A1$
11. **Elitism:** Perform Pareto ranking and crowding distance to $Pop(g) \cup A1$. Select the best N solutions to form new population $Pop(g)$
12. **PSO:** Perform PSO update to all solutions in $Pop(g)$. Store the offspring in $A2$
13. **Evaluation:** Evaluate all offspring in $A2$
14. **Archiving:** Combine parent and offspring $Pop(g) \cup A2$
15. **Elitism:** Perform Pareto ranking and crowding distance to $Pop(g) \cup A2$. Select the best N solutions to form new population $Pop(g+1)$. $g = g + 1$

End Do

16. **Output:** Output the final set of solutions $Pop(g)$

End

Figure 5.1: Pseudo-code of PLREDA

number representation is direct [163]. The first discrete PSO was developed by Kennedy and Eberhart [164]. In binary PSO, each particle represents a position in binary space. Cedeno and Agrafiotis [165] implemented a binary PSO in feature selection by treating the position vectors as continuous-valued vectors. The continuous-valued vectors are then transformed to discrete-valued vectors by setting a threshold to decide between the bit-values 0 and 1.

In this implementation, the velocity $(v_{k,d}(g))$ of each d^{th} dimension of particle k in generation g is updated according to the continuous-valued case as:

$$v_{k,d}(g+1) = \alpha v_{k,d}(g) + c_1 r_1 \left(pbest_{k,d}(g) - x_{k,d}(g)\right) + c_2 r_2 \left(gbest_d(g) - x_{k,d}(g)\right) \quad (5.2)$$

where $v_{k,d} \in [vmax_d, vmin_d]$, $vmax_d$ and $vmin_d$ are the maximum and minimum velocity in the d^{th} dimension, respectively. $gbest_d(g)$ is the location in d^{th} dimension parameter space of the best fitness returned for the entire swarm in generation g. In this implementation, tournament selection is applied to find the best individual at the current iteration. The best individual is the

one that is non-dominated and less crowded by other solutions. $pbest_{k,d}(g)$ is the location in the d^{th} dimension parameter space of the best fitness returned for a particle k in generation g. The factors r_1 and r_2 are uniformly distributed random variables in the range of [0, 1], while c_1 and c_2 are the weights that regulate the influences between global and local information. Another factor α, known as initial weight, aims to balance the global and local search abilities. After updating the velocity, the position of each particle is updated according to the equations as follows:

$$y_{k,d}(g+1) = y_{k,d}(g) + v_{k,d}(g+1) \tag{5.3}$$

$$x_{k,d}(g+1) = \begin{cases} 1 & \text{if } y_{k,d}(g+1) \geq Threshold \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

where $y_{k,d}$ is the updated position in the d^{th} dimension of particle k in a continuous domain while $x_{k,d}$ is the corresponding position values in a discrete domain after transformation by a reference threshold. The threshold is evolutionary and is treated as a variable to be optimized.

5.3.3 Probability Dominance

The concept of probability dominance was proposed in [146]. In PLREDA, this concept is implemented to determine the probability error of each selected individual. This information is used to group the population into several clusters before a probabilistic model is built.

Assume that $f_i(A)$ and $f_i(B)$ are two solutions in the objective space with m objective functions ($i = 1, 2, ..., m$). In a noise free situation, $f_i(A)$ is said to strictly dominate $f_i(B)$ if $f_i(A)$ is smaller than $f_i(B)$ in all the objective values in minimization case. On the other hand, $f_i(A)$ and $f_i(B)$ are mutually non-dominated only if not all the objective values in one solution are lower than that of the other. Figure 5.2 further illustrates the concept of dominance. In the figure, point Z is strictly dominated by point Y, while points X and Y are mutually non-dominated.

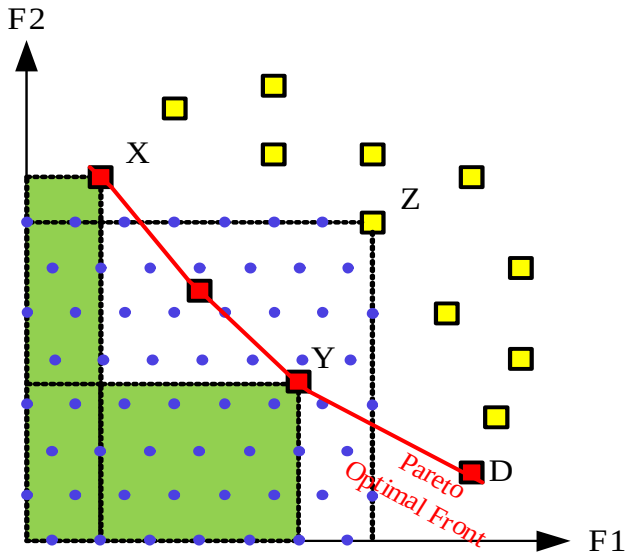


Figure 5.2: Concept of dominance

In a noisy domain, the above statements may not correctly represent the true domination behaviour. Even through $f_i(A)$ appears to strictly dominate $f_i(B)$, the noise may distort the actual fitness values where $f_i(B)$ is supposed to dominate $f_i(A)$. In the selection process, the selection error occurs when the less fit individual is chosen. Therefore, the probability to make an error in the selection process could be utilized to improve the decision making process.

In [146], Hughes suggested a probabilistic selection model, which employs the probabilistic dominance to account for the noise effect in the objective space. The noise is assumed to be normally distributed. In this model, $P(f_i(A) < f_i(B))$ is the probability error that $f_i(A)$ dominates $f_i(B)$ and it is calculated as:

$$P\left(f_i\left(A\right) < f_i\left(B\right)\right) = 0.5 + 0.5 \operatorname{erf}\left(\frac{f_i(A) - f_i(B)}{2\sigma_n}\right) \tag{5.5}$$

where σ_n is the standard deviation of the distribution of the noise, and $\operatorname{erf}(x)$ is the error function. However, the calculation of the error function is time consuming. Therefore, the probability error (P_e) is approximated as follows:

$$P_{e,i} \equiv P\left(f_i\left(A\right) < f_i\left(B\right)\right) \approx 0.5 + 0.5 \tanh\left(\frac{f_i(A) - f_i(B)}{1.6\sigma_n}\right) \tag{5.6}$$

For m objectives, the final probability (P_e) that solution A dominates solution B is calculated as follows.

$$P_e = P\left(F\left(A\right) < F\left(B\right)\right) = \prod_{i=1}^{m} P\left(f_i(A) < f_i(B)\right) \tag{5.7}$$

5.3.4 Likelihood Correction

In a noisy condition, the selected candidate solutions may not represent the best solutions (best in terms of lower Pareto rank or larger crowding distance). Therefore, the probabilistic model built by the REDA may not correctly represent the real distribution of the best solutions. In order to improve the appropriate probabilistic model, likelihood correction is proposed. This correction is based on the heuristic that if the distribution can be approximated as close to the real distribution of the best solutions, the detrimental effect of the noise can then reduced. To approximate the real distribution, the probability of making an error in the selection process is adapted in the probabilistic modelling. In binary tournament selection, if two solutions in the tournament have a huge distinction in their objective values, for example $f_i(A)$ dominates $f_i(B)$ by far, then the selection error for selecting $f_i(A)$ is small. On the other hand, if two individuals in the tournament are near to each other in the objective space, then the selection error is larger. Therefore, if the probabilistic model built by the REDA is only based on individuals with small selection errors, then, the model may avoid distortions caused by those solutions with large selection errors.

However, the probability distribution may not come close to the real distribution if the number of individuals with smaller selection errors is too little. Thus, a method to combine the distribution between solutions with small selection errors and those with large selection errors

is designed. This combination is based on the penalty approach where individuals with smaller selection errors will be penalized less while solutions with larger selection errors will be more heavily penalized. This is because the real distribution is more likely to follow the distribution of the population with smaller selection errors than those with larger selection errors.

Thus, the first step is to determine the number of groups (Gp) and then sort the individuals into Gp clusters according to the predefined threshold ($Th_y, y = \{1, 2, ..., Gp-1\}$) of the probability error (P_e) according to equation (5.8).

$$x_i = \begin{cases} 1 & \text{if } 0 \leq P_e < Th_1 & \text{then } P_e = 0.0 \\ 2 & \text{if } Th_1 \leq P_e < Th_2 & \text{then } P_e = Th_1 \\ \vdots & \vdots & \vdots \\ Gp & \text{if } Th_{Gp-1} \leq P_e < 1 & \text{then } P_e = Th_{Gp-1} \end{cases} \tag{5.8}$$

P_e is calculated by using equation (5.7). Th_y, the threshold, is determined through experimental investigation. In equation (5.8), the function of Th_y is to define the criterion for grouping. For example, solutions with P_e less than Th_1 will be clustered into the first group. Solutions with P_e in between Th_1 and Th_2 will be clustered into the second group, and so forth. Afterward, the P_e is re-determined to be the value of the threshold (penalty value). Then, the distribution in each cluster is combined according to equation (5.9).

$$p_g(v_i = 1) = \frac{\sum_{l=1}^{N} \delta_l(v_i^+)(1 - P_e) + avg\left(\sum_{l=1}^{N} \delta_l(v_i)\right)}{\sum_{l=1}^{N} \delta_l(v_i^+)(1 - P_e) + \sum_{l=1}^{N} \delta_l(v_i^-)(1 - P_e) + r_i \times avg\left(\sum_{l=1}^{N} \delta_l(v_i)\right)} \tag{5.9}$$

In equation (5.9), the penalty assigned to each cluster is determined by the threshold of the probability error of the members in the cluster. The probability error P_e in cluster 1 is always treated as 0.0 because cluster 1 has the smallest selection error than other clusters, thus, the probability distribution contributed by the solutions in this cluster will not be penalized. The solutions in

other clusters will be penalized (according to the threshold value) since these solutions may not be the fittest ones in the population.

5.4 Problem Description and Implementation

This section presents the test functions and the experimental settings of the implementation. In the experimental studies, the assumption is made that there is advance knowledge of the presence of noise.

5.4.1 Noisy Test Problems

Eight benchmark test instances, including ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, DTLZ1, ZDTL2, and DTLZ3, are used to test the performance of the PLREDA in terms of converging to the true Pareto optimal front and maintaining a set of diverse solutions. All of the test functions have different characteristics and shapes of the Pareto optimal front. ZDT problems [101] consist two objectives functions while DTLZ problems [102] possess three objective functions. Detailed information of these test problems can be referred to Section 2.5.

5.4.2 Implementation

Three performance indicators, GD, MS, and IGD, are used to show the numerical comparison between the PLREDA and other state-of-the-art algorithms. Detailed description of these performance metrics can be referred to Section 2.4.

Seven MOEAs were chosen for performance comparison with the PLREDA. Since PLREDA is developed based on REDA, REDA should be included in the comparison. Likelihood correction is introduced into REDA and forms LREDA. This algorithm could show the advantages of incorporating the likelihood correction. Other than that, five state-of-the-art MOEAs are also chosen for performance comparison. Firstly, NSGA-II [32] is a popular algorithm in evolutionary multi-objective optimization for its ability to generate promising solutions for most of the

test problems. The re-sampling mechanism is incorporated into NSGA-II and the algorithm is called RNSGA-II. Next, NTSPEA [155] and MOPSEA [146] are algorithms which were specifically proposed to handle noisy objective functions. The fundamental noise handling principle in MOPSEA is based on the probability dominance. Finally, MOEARF [46] is a recently published noise handling MOEA.

All algorithms were implemented in C++ and ran on an Intel Core 2 Duo, 3.0 GHz personal computer. The experimental settings are presented in Table 5.1. The parameter settings were determined through preliminary experimental studies or taken from previous work.

Table 5.1: Parameter settings

Parameter	Setting
Representation	Binary-number representation with 15 bits per decision variable
Population size	100 in ZDT problems and 200 in DTLZ problems
Archive size	100 in ZDT problems and 200 in DTLZ problems
Selection	Binary tournament selection
Fitness evaluations	40,000 in ZDT problems and 80,000 in DTLZ problems
Noise levels	0%, 5%, 10%, 20%
Independent runs	30
Crossover rate in NSGA-II, RNSGA-II, NTSPEA, MOPSEA, and MOEARF	0.8
Mutation rate in NSGA-II, RNSGA-II, NTSPEA, MOPSEA, and MOEARF	1/(Variable_size $\times$ Variable_bit)
Re-sampling in RNSGA-II	4
Number of hidden units in REDA, LREDA, and PLREDA	10 in ZDT problems and 5 in DTLZ1 problems
Number of training epochs in REDA, LREDA, and PLREDA	20 in all test problems
Number of groups in LREDA and PLREDA	3
Thresholds (Th_y) in LREDA and PLREDA	$Th_1 = 0.25, Th_2 = 0.5$
α in PLREDA	0.99
c_1 and c_2 in PLREDA	2 and 1.5
$vmax$ and $vmin$ in PLREDA	1.0 and -1.0

5.5 Results and Discussions

In this section, the studies are divided into four categories. The comparison results of the algorithms are presented in the first category. This is followed by the analysis of the scalability behaviour of the algorithms. The possibility of hybridization with other evolutionary paradigms is discussed next. Finally, the computational time for the algorithms to perform a single simulation run is examined.

5.5.1 Comparison Results

Table 5.2: GD for ZDT1-ZDT4 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT1(30)-0%	ZDT1(30)-5%	ZDT1(30)-10%	ZDT1(30)-20%
PLREDA	0.0045±0.0016	0.0598±0.0351	**0.0760±0.0339**	**0.1520±0.0697**
LREDA	0.0038±0.0003	0.0689±0.0117	0.1319±0.0188	0.2400±0.0511
REDA	0.0038±0.0003	0.0711±0.0125	0.1341±0.0245	0.2650±0.0487
NSGA-II	0.0038±0.0003	0.1156±0.0180	0.1966±0.0325	0.3323±0.0469
NTSPEA	0.0063±0.0006	0.1116±0.0174	0.1949±0.0324	0.3351±0.0498
MOPSEA	**0.0035±0.0003**	0.0968±0.0138	0.1493±0.0255	0.2464±0.0340
RNSGA-II	0.0038±0.0003	0.1003±0.0158	0.1586±0.0278	0.2769±0.0533
MOEARF	0.0039±0.0004	**0.0468±0.0052**	0.0948±0.0136	0.1769±0.0213
Algorithm	ZDT2(30)-0%	ZDT2(30)-5%	ZDT2(30)-10%	ZDT2(30)-20%
PLREDA	0.0039±0.0005	**0.0429±0.0423**	**0.0614±0.0380**	**0.1386±0.1230**
LREDA	0.0038±0.0002	0.0996±0.0174	0.2036±0.0371	0.4360±0.0726
REDA	0.0038±0.0002	0.1081±0.0234	0.1942±0.0411	0.4412±0.0833
NSGA-II	0.0038±0.0002	0.1513±0.0292	0.2833±0.0542	0.5225±0.0937
NTSPEA	0.0063±0.0035	0.1296±0.0245	0.2563±0.0422	0.5042±0.1078
MOPSEA	**0.0036±0.0002**	0.1084±0.0172	0.1916±0.0261	0.3875±0.0553
RNSGA-II	0.0038±0.0002	0.1757±0.0359	0.2694±0.0609	0.4493±0.0892
MOEARF	0.0042±0.0005	0.0494±0.0063	0.0971±0.0137	0.2218±0.0387
Algorithm	ZDT3(30)-0%	ZDT3(30)-5%	ZDT3(30)-10%	ZDT3(30)-20%
PLREDA	0.0093±0.0017	**0.0331±0.0104**	**0.0558±0.0383**	**0.1228±0.0688**
LREDA	0.0086±0.0008	0.0413±0.0050	0.0704±0.0204	0.2375±0.0299
REDA	0.0086±0.0008	0.0393±0.0060	0.0708±0.0209	0.2299±0.0353
NSGA-II	0.0086±0.0008	0.0553±0.0112	0.1129±0.0404	0.2739±0.0545
NTSPEA	0.0096±0.0009	0.0595±0.0177	0.1277±0.0471	0.2709±0.0488
MOPSEA	**0.0064±0.0006**	0.0499±0.0097	0.0767±0.0245	0.1289±0.0703
RNSGA-II	0.0086±0.0023	0.0782±0.0151	0.0993±0.0250	0.2348±0.0435
MOEARF	0.0087±0.0011	0.0463±0.0077	0.0744±0.0211	0.1764±0.0553
Algorithm	ZDT4(10)-0%	ZDT4(10)-5%	ZDT4(10)-10%	ZDT4(10)-20%
PLREDA	0.5130±1.6893	**0.0047±0.0010**	**0.0042±0.0014**	**0.0066±0.0047**
LREDA	0.5332±0.2244	0.4658±0.2169	0.5942±0.2827	0.6040±0.2204
REDA	0.5332±0.2244	0.5324±0.1717	0.5405±0.1999	0.6232±0.2372
NSGA-II	0.5169±0.1978	0.5432±0.2086	0.5857±0.1945	0.6090±0.2147
NTSPEA	0.5777±0.1847	0.5533±0.2268	0.5538±0.2046	0.5826±0.2264
MOPSEA	0.5493±0.2190	0.5131±0.1883	0.5826±0.2264	0.5939±0.2307
RNSGA-II	0.5169±0.1679	0.5686±0.1650	0.5927±0.1889	0.6701±0.2649
MOEARF	**0.0148±0.0034**	0.0105±0.0028	0.0262±0.0106	0.4861±1.0196

Tables 5.2 and 5.3 show the results in terms of the GD measurement obtained from the different algorithms. The mean and standard deviation of the results over 30 independent runs are presented. The best result in each test instance is highlighted in bold. It is obvious from the tabulation that PLREDA is able to obtain the best GD results in most of the test instances, followed by MOEARF and MOPSEA. In a noiseless condition, MOPSEA has obtained the best results in four of the test instances, with MOEARF and PLREDA obtaining two best results respectively. When the noise levels are increased, PLREDA outperformed other algorithms except for DTLZ2. In addition, we can see the improvement from REDA to LREDA, in most of the test instances, with noisy condition. The performance of LREDA is further enhanced when it is hybridized with PSO (PLREDA). It was also observed that the hybridization between LREDA and

Table 5.3: GD for ZDT6, DTLZ1-DTLZ3 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT6(10)-0%	ZDT6(10)-5%	ZDT6(10)-10%	ZDT6(10)-20%
PLREDA	0.0130±0.0010	**0.0101±0.0027**	**0.0081±0.0011**	**0.0070±0.0021**
LREDA	0.0128±0.0010	0.0623±0.1702	0.2378±0.3061	0.8045±0.4053
REDA	0.0128±0.0010	0.0717±0.1682	0.2924±0.3158	0.8379±0.5210
NSGA-II	0.0125±0.0011	0.0252±0.0016	0.3219±0.3582	1.3716±0.2406
NTSPEA	0.0426±0.1194	0.0250±0.0026	0.6900±0.2574	1.4351±0.2195
MOPSEA	0.3706±0.7834	0.7933±0.1971	1.0991±0.1726	1.7154±0.2660
RNSGA-II	0.0125±0.0011	0.9842±0.1275	1.1682±0.1895	1.5529±0.1691
MOEARF	**0.0101±0.0043**	0.0635±0.1391	0.6011±0.3988	1.0745±0.3266
Algorithm	DTLZ1(30)-0%	DTLZ1(30)-5%	DTLZ1(30)-10%	DTLZ1(30)-20%
PLREDA	**57.701±95.420**	**238.91±77.411**	**245.66±62.950**	**275.41±472.80**
LREDA	204.73±50.023	345.50±71.277	399.83±97.612	466.79±109.74
REDA	204.73±50.023	377.54±70.179	457.16±83.127	472.80±79.401
NSGA-II	122.80±35.134	698.10±215.22	849.50±198.22	913.30±217.38
NTSPEA	128.60±38.403	800.40±176.85	777.20±151.10	840.10±180.99
MOPSEA	1066.2±347.89	1051.3±173.33	1004.8±203.57	1034.9±178.07
RNSGA-II	122.80±35.134	1484.4±95.267	1510.6±83.930	1551.4±66.218
MOEARF	486.70±138.86	1237.0±269.63	1298.0±179.37	1249.4±253.14
Algorithm	DTLZ2(30)-0%	DTLZ2(30)-5%	DTLZ2(30)-10%	DTLZ2(30)-20%
PLREDA	0.0664±0.0069	0.1425±0.0237	0.2224±0.0591	0.4682±0.0775
LREDA	0.1044±0.0123	0.2045±0.0345	0.2968±0.0654	0.4873±0.1003
REDA	0.1044±0.0123	0.2124±0.0416	0.3112±0.0591	0.4911±0.0804
NSGA-II	0.0629±0.0033	0.1622±0.0393	0.2663±0.0729	0.5195±0.0969
NTSPEA	0.0757±0.0044	0.1645±0.0641	0.2397±0.0667	0.6225±0.1999
MOPSEA	**0.0530±0.0068**	**0.1029±0.0199**	**0.1471±0.0230**	**0.2709±0.0326**
RNSGA-II	0.0629±0.0033	0.4997±0.1680	0.5134±0.1353	0.6589±0.1325
MOEARF	0.0777±0.0127	0.2906±0.0586	0.3949±0.1275	0.6032±0.1373
Algorithm	DTLZ3(30)-0%	DTLZ3(30)-5%	DTLZ3(30)-10%	DTLZ3(30)-20%
PLREDA	**94.175±137.75**	**427.72±109.86**	**537.46±160.78**	**515.01±156.10**
LREDA	224.99±54.689	541.58±95.543	566.62±130.31	570.19±130.86
REDA	224.99±54.689	554.49±145.25	653.08±112.36	589.37±146.69
NSGA-II	128.50±35.946	651.00±251.07	608.90±177.56	653.60±218.42
NTSPEA	266.00±252.70	1075.5±310.33	942.40±236.77	1071.4±307.64
MOPSEA	1493.2±295.54	1111.0±171.45	1109.2±163.71	1047.4±193.59
RNSGA-II	128.50±35.946	1684.2±105.79	1700.4±120.75	1644.9±106.64
MOEARF	571.60±336.39	935.00±193.19	898.20±179.99	809.20±339.23

PSO may affect the search ability in some noiseless circumstances (ZDT1, ZDT2, and ZDT3). However, the performance of the PLREDA was outstanding in noisy environments. Since GD measures the distance between the evolved Pareto front and the Pareto optimal front, the closer they are, the smaller the GD value. It was observed that all of the algorithms are able to obtain a set of solutions that is near to the Pareto optimal front in ZDT1, ZDT2, ZDT3, ZDT6, and DTLZ2. For the rest of the test problems (ZDT4, DTLZ1, and DTLZ3), most of the algorithms are trapped in local optima. This is attributed to the fact that three of these test problems consisted of many local optimal fronts. MOEARF was able to generate good solutions in ZDT4; however, its performance is poor in DTLZ1 and DTLZ3. PLREDA performed better than MOEARF in all DTLZ problems, even though the algorithm also failed to reach the Pareto optimal front.

The good performance of PLREDA was due to the combination of REDA, PSO, and the

likelihood correction feature. First of all, REDA is more robust than NSGA-II because the performance of REDA is better than NSGA-II in noisy conditions. REDA, which performs the search by modelling the global probability distribution of the population, is more responsive since the reproduction is based on the global information and not individual solutions. Furthermore, likelihood correction is able to tune the probability distribution so that the distribution of the solutions is more likely to follow the one with a smaller selection error. The hybridization has further enhanced the ability of REDA in exploring the search space, especially in noisy conditions. This hybridization is utterly important when the REDA fails to model the promising regions in the search space. In that case, the hybridization could provide opportunities to explore those regions, thus, improving the search ability. Re-sampling is probably one of the simplest noise handling mechanisms, which has been implemented in RNSGA-II. The performance of this algorithm is better than NSGA-II in ZDT problems. However, its performance is poor in DTLZ problems. Even though re-sampling is able to reduce the effect of noise, it incurred extra fitness evaluations. Since the fitness evaluation in DTLZ problems is limited to 80,000, RNSGA-II failed to converge at the end of the evolution. NTSPEA and MOPSEA, which are noise handling algorithms, have shown better results in noisy environments compared to NSGA-II. This is unquestionable since NTSPEA and MOPSEA tackled the noisy information by incorporating the domination dependent lifetime and probability dominance respectively, while NSGA-II did not take any extra measurements into account when handling the noisy data. For MOEARF, the noise handling mechanism is based on posterior knowledge of the favourable movements, thus it was able to obtain good results in ZDT problems. However, the performance of MOEARF in DTLZ problems is poorer than those of all other algorithms except RNSGA-II.

Tables 5.4 and 5.5 show the results in terms of MS measurement obtained from the different algorithms. Generally, PLREDA was able to evolve a set of most diverse solutions in most of the test instances, followed by MOEARF and MOPSEA. The incorporation of likelihood correction was also able to improve the diversity of REDA in some of the test instances. There was no clear

Table 5.4: MS for ZDT1-ZDT4 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT1(30)-0%	ZDT1(30)-5%	ZDT1(30)-10%	ZDT1(30)-20%
PLREDA	**0.9982±0.0014**	0.9406±0.0281	0.9130±0.0399	0.8731±0.0447
LREDA	0.9928±0.0016	0.9263±0.0230	0.8856±0.0358	0.8319±0.0582
REDA	0.9928±0.0016	0.9251±0.0217	0.8998±0.0303	0.8297±0.0437
NSGA-II	0.9979±0.0007	0.9133±0.0267	0.8714±0.0445	0.8128±0.0432
NTSPEA	0.9970±0.0007	0.9116±0.0229	0.8561±0.0362	0.7693±0.0464
MOPSEA	0.9881±0.0045	0.9162±0.0297	0.8535±0.0546	0.7894±0.0586
RNSGA-II	0.9979±0.0009	0.9264±0.0292	0.8904±0.0308	0.8399±0.0322
MOEARF	0.9768±0.0144	**0.9680±0.0161**	**0.9380±0.0161**	**0.8949±0.0432**
Algorithm	ZDT2(30)-0%	ZDT2(30)-5%	ZDT2(30)-10%	ZDT2(30)-20%
PLREDA	**0.9973±0.0023**	0.9073±0.0451	0.8629±0.0643	0.7056±0.1237
LREDA	0.9877±0.0011	0.8646±0.0301	0.7517±0.0410	0.4699±0.1898
REDA	0.9877±0.0011	0.8513±0.0277	0.7478±0.0548	0.4778±0.1370
NSGA-II	0.9934±0.0027	0.8300±0.0408	0.7318±0.0665	0.5086±0.2445
NTSPEA	0.9901±0.0072	0.8022±0.0370	0.5340±0.1602	0.1102±0.1428
MOPSEA	0.9817±0.0072	0.8022±0.0370	0.5340±0.1602	0.1102±0.1428
RNSGA-II	0.9934±0.0027	0.7865±0.0449	0.6632±0.1516	0.4784±0.2083
MOEARF	0.9812±0.0118	**0.9565±0.0082**	**0.9169±0.0187**	**0.7412±0.2345**
Algorithm	ZDT3(30)-0%	ZDT3(30)-5%	ZDT3(30)-10%	ZDT3(30)-20%
PLREDA	0.9987±0.0022	**0.9707±0.0158**	**0.9566±0.0215**	**0.9190±0.0550**
LREDA	0.9985±0.0007	0.9602±0.0106	0.9409±0.0136	0.8654±0.0677
REDA	0.9985±0.0007	0.9625±0.0106	0.9418±0.0131	0.8821±0.0575
NSGA-II	**0.9998±0.0001**	0.8973±0.0370	0.8760±0.0196	0.8117±0.0561
NTSPEA	0.9862±0.0240	0.8871±0.0391	0.8226±0.0668	0.7517±0.0836
MOPSEA	0.9753±0.0415	0.8901±0.0557	0.7706±0.1480	0.6446±0.1646
RNSGA-II	**0.9998±0.0001**	0.8916±0.0384	0.8615±0.0824	0.8284±0.0643
MOEARF	0.9712±0.0537	0.9418±0.0061	0.9265±0.0138	0.8775±0.0587
Algorithm	ZDT4(10)-0%	ZDT4(10)-5%	ZDT4(10)-10%	ZDT4(10)-20%
PLREDA	0.9302±1.1159	0.9372±0.0217	0.9168±0.0221	0.8575±0.0653
LREDA	0.7774±0.0714	0.7569±0.0820	0.7117±0.0665	0.6865±0.0700
REDA	0.7774±0.0714	0.7339±0.0596	0.7171±0.0707	0.6762±0.0554
NSGA-II	0.7545±0.0681	0.7469±0.0642	0.7337±0.0626	0.7084±0.0765
NTSPEA	0.7352±0.0500	0.7394±0.0710	0.7179±0.0571	0.6599±0.0680
MOPSEA	0.7176±0.0716	0.7328±0.0618	0.6921±0.0720	0.6344±0.0903
RNSGA-II	0.7545±0.0681	0.7358±0.0468	0.7156±0.0494	0.6921±0.0663
MOEARF	**0.9544±0.0182**	**0.9769±0.0175**	**0.9648±0.0303**	**0.8692±0.0944**

difference in performance between REDA and NSGA-II. Furthermore, re-sampling was unable to maintain or improve the diversification of NSGA-II. It was also observed that the noise interfered with the diversity performance of all algorithms in all ZDT problems. For DTLZ problems, the negative influence of noise towards the diversity preservation in all algorithms was far smaller than for ZDT problems.

Among the algorithms, PLREDA, LREDA, REDA, NSGA-II, and RNSGA-II maintained the diversity of the solutions through crowding measurement while NTSPEA, MOPSEA, and MOEARF kept the diversity of the solutions by using a niche sharing mechanism. There was not much difference between both of the diversity preservation mechanisms in both noisy and noiseless environments. In fact, both of the methods are able to maintain a set of diverse solutions even though the convergence was not good, which is indicated by the MS values being near to 1.0.

Table 5.5: MS for ZDT6, DTLZ1-DTLZ3 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT6(10)-0%	ZDT6(10)-5%	ZDT6(10)-10%	ZDT6(10)-20%
PLREDA	**1.0000±0.0000**	**0.9993±0.0015**	**0.9990±0.0017**	**0.9982±0.0027**
LREDA	**1.0000±0.0000**	0.9879±0.0535	0.9336±0.1209	0.7751±0.1258
REDA	**1.0000±0.0000**	0.9904±0.0485	0.8976±0.1360	0.7838±0.1319
NSGA-II	**1.0000±0.0000**	0.9946±0.0013	0.8769±0.1457	0.6983±0.0048
NTSPEA	0.9894±0.0520	0.9925±0.0039	0.7289±0.0897	0.6921±0.0204
MOPSEA	0.9223±0.1292	0.6929±0.0371	0.6925±0.0182	0.6547±0.1701
RNSGA-II	0.9232±0.1217	0.6999±0.0006	0.6967±0.0102	0.6988±0.0027
MOEARF	0.9963±0.0128	0.9948±0.0005	0.9947±0.0014	0.9939±0.0030
Algorithm	DTLZ1(30)-0%	DTLZ1(30)-5%	DTLZ1(30)-10%	DTLZ1(30)-20%
PLREDA	**0.9980±0.0037**	**0.9927±0.0040**	**0.9905±0.0069**	**0.9876±0.0080**
LREDA	0.9944±0.0024	0.9850±0.0087	0.9790±0.0168	0.9688±0.0291
REDA	0.9944±0.0024	0.9828±0.0200	0.9755±0.0167	0.9743±0.0160
NSGA-II	0.9934±0.0027	0.8300±0.0408	0.7318±0.0665	0.5086±0.2445
NTSPEA	0.9956±0.0048	0.9850±0.0085	0.9836±0.0097	0.9794±0.0084
MOPSEA	0.9816±0.0067	0.9688±0.0140	0.9630±0.0173	0.9554±0.0208
RNSGA-II	0.9959±0.0025	0.9701±0.0037	0.9684±0.0042	0.9668±0.0049
MOEARF	0.9716±0.0094	0.9579±0.0091	0.9543±0.0114	0.9529±0.0164
Algorithm	DTLZ2(30)-0%	DTLZ2(30)-5%	DTLZ2(30)-10%	DTLZ2(30)-20%
PLREDA	**1.0000±0.0000**	0.9989±0.0001	0.9987±0.0008	0.9978±0.0023
LREDA	**1.0000±0.0000**	0.9986±0.0001	0.9978±0.0014	0.9977±0.0016
REDA	**1.0000±0.0000**	0.9985±0.0001	0.9982±0.0010	0.9979±0.0011
NSGA-II	**1.0000±0.0000**	0.9995±0.0004	0.9990±0.0008	0.9983±0.0017
NTSPEA	0.9993±0.0008	0.9990±0.0007	0.9989±0.0008	0.9983±0.0015
MOPSEA	0.9998±0.0009	0.9985±0.0014	0.9982±0.0018	0.9980±0.0013
RNSGA-II	**1.0000±0.0000**	0.9997±0.0002	0.9994±0.0005	0.9993±0.0005
MOEARF	**1.0000±0.0000**	**0.9999±0.0002**	**0.9998±0.0003**	**0.9996±0.0003**
Algorithm	DTLZ3(30)-0%	DTLZ3(30)-5%	DTLZ3(30)-10%	DTLZ3(30)-20%
PLREDA	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**
LREDA	0.9994±0.0013	0.9790±0.0309	0.9827±0.0215	0.9743±0.0205
REDA	0.9994±0.0013	0.9835±0.0284	0.9814±0.0707	0.6762±0.0554
NSGA-II	**1.0000±0.0000**	0.9994±0.0011	0.9993±0.0008	0.9971±0.0043
NTSPEA	0.9996±0.0018	0.9977±0.0029	0.9986±0.0015	0.9979±0.0022
MOPSEA	0.9998±0.0003	0.9981±0.0024	0.9963±0.0036	0.9933±0.0061
RNSGA-II	**1.0000±0.0000**	0.9989±0.0022	0.9980±0.0048	0.9973±0.0055
MOEARF	0.9998±0.0011	0.9979±0.0027	0.9961±0.0049	0.9961±0.0038

In DTLZ problems, no matter how far are the evolved solutions from optimality, all algorithms were successful in maintaining a set of diverse solutions ($MS > 0.9$). This may be due to the fact that the number of non-dominated solutions in DTLZ problems is much more than those in ZDT problems since DTLZ problems consist of three objective functions while ZDT problems only have two objective functions. In addition, it may be also caused by the setting of the population size.

Tables 5.6 and 5.7 show the results in terms of IGD measurement obtained from the different algorithms. PLREDA showed the best performance in most of the test instances, followed by MOEARF. The performance of REDA is improved in most of the test instances when likelihood correction is incorporated (LREDA). Comparing REDA and NSGA-II, algorithms without any noise handling features, REDA is more robust than NSGA-II. This is shown by a lower value of

Table 5.6: IGD for ZDT1-ZDT4 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT1(30)-0%	ZDT1(30)-5%	ZDT1(30)-10%	ZDT1(30)-20%
PLREDA	0.0054±0.0015	0.0547±0.0288	**0.0684±0.0278**	**0.1178±0.0512**
LREDA	**0.0042±0.0002**	0.0633±0.0099	0.1139±0.0138	0.1960±0.0421
REDA	**0.0042±0.0002**	0.0659±0.0109	0.1128±0.0212	0.2087±0.0392
NSGA-II	0.0047±0.0002	0.1101±0.0165	0.1912±0.0275	0.3239±0.0461
NTSPEA	0.0073±0.0006	0.1092±0.0165	0.1912±0.0275	0.3239±0.0461
MOPSEA	0.0044±0.0004	0.0966±0.0125	0.1544±0.0264	0.2486±0.0473
RNSGA-II	0.0047±0.0002	0.0946±0.0155	0.1475±0.0261	0.2448±0.0473
MOEARF	0.0098±0.0016	**0.0460±0.0053**	0.0841±0.0110	0.1492±0.0212
Algorithm	ZDT2(30)-0%	ZDT2(30)-5%	ZDT2(30)-10%	ZDT2(30)-20%
PLREDA	0.0048±0.0004	0.0502±0.0349	**0.0724±0.0341**	**0.1680±0.1202**
LREDA	**0.0043±0.0001**	0.0934±0.0168	0.1882±0.0338	0.4973±0.1511
REDA	**0.0043±0.0001**	0.1009±0.0214	0.1834±0.0419	0.4808±0.1330
NSGA-II	0.0047±0.0003	0.1529±0.0322	0.2740±0.0633	0.6573±0.2060
NTSPEA	0.0089±0.0034	0.1367±0.0256	0.2688±0.0421	0.6436±0.2214
MOPSEA	0.0048±0.0003	0.1295±0.0195	0.3040±0.1019	0.7329±0.1145
RNSGA-II	0.0047±0.0003	0.1800±0.0372	0.3011±0.1052	0.5454±0.2075
MOEARF	0.0098±0.0012	**0.0500±0.0051**	0.0877±0.0117	0.2237±0.1476
Algorithm	ZDT3(30)-0%	ZDT3(30)-5%	ZDT3(30)-10%	ZDT3(30)-20%
PLREDA	0.0062±0.0030	**0.0423±0.0160**	**0.0943±0.0408**	**0.1575±0.0533**
LREDA	0.0058±0.0007	0.0769±0.0096	0.1225±0.0197	0.2299±0.0332
REDA	0.0058±0.0007	0.0744±0.0150	0.1245±0.0164	0.2222±0.0324
NSGA-II	**0.0053±0.0004**	0.1097±0.0162	0.1739±0.0216	0.3012±0.0552
NTSPEA	0.0104±0.0055	0.1225±0.0219	0.2048±0.0260	0.3510±0.0534
MOPSEA	0.0128±0.0072	0.1028±0.0149	0.1884±0.0433	0.3287±0.0585
RNSGA-II	**0.0053±0.0004**	0.1089±0.0154	0.1580±0.0333	0.2333±0.0389
MOEARF	0.0154±0.0078	0.0571±0.0085	0.1027±0.0169	0.1838±0.0260
Algorithm	ZDT4(10)-0%	ZDT4(10)-5%	ZDT4(10)-10%	ZDT4(10)-20%
PLREDA	0.3898±0.9901	**0.0157±0.0042**	**0.0291±0.0088**	**0.0581±0.0205**
LREDA	0.5403±0.2386	0.4714±0.2358	0.5541±0.2277	0.5746±0.2268
REDA	0.5403±0.2386	0.5346±0.1796	0.5382±0.2145	0.5845±0.2193
NSGA-II	0.5060±0.2019	0.5509±0.2257	0.5870±0.2142	0.6021±0.2351
NTSPEA	0.5663±0.1926	0.5757±0.2513	0.5680±0.2157	0.6097±0.1778
MOPSEA	0.5405±0.2256	0.5241±0.2027	0.6020±0.2399	0.6358±0.2493
RNSGA-II	0.5853±0.1753	0.5748±0.1791	0.5994±0.2057	0.6584±0.2521
MOEARF	**0.0108±0.0018**	0.0192±0.0029	0.0332±0.0070	0.1330±0.0884

IGD in REDA. However, the robustness of the REDA is not due to the type of Gaussian noise implemented in this chapter (Gaussian noise with zero mean). This is because REDA does not take any average information from the population. Instead, it measures the probability of existence of any cardinalities in the decision variables. As with the observation in Table 2, re-sampling in RNSGA-II has improved the IGD value of NSGA-II in ZDT problems but the performance is poor in DTLZ problems. For MOEARF, the convergence and diversity maintenance were good in ZDT problems; however, the performance was poor in DTLZ problems.

Figure 5.3 shows the Pareto front of ZDT3 generated from the different algorithms. The solutions plotted are the non-dominated solutions obtained from all 30 simulation runs. It is clear that all of the algorithms were able to maintain a set of good diverse solutions. In terms of convergence, PLREDA and MOEARF have an evolved Pareto front which is closed to the

Table 5.7: IGD for ZDT6, DTLZ1-DTLZ3 under the influences of different noise levels

Algorithm	Problem(n)-Noise			
	ZDT6(10)-0%	ZDT6(10)-5%	ZDT6(10)-10%	ZDT6(10)-20%
PLREDA	**0.0027±0.0003**	0.0161±0.0045	**0.0229±0.0096**	**0.0365±0.0154**
LREDA	**0.0027±0.0003**	0.0451±0.1344	0.1984±0.3230	0.8468±0.5366
REDA	**0.0027±0.0003**	0.0391±0.1046	0.2901±0.3714	0.8846±0.5840
NSGA-II	0.0029±0.0003	0.0104±0.0026	0.3722±0.4578	1.6392±0.2960
NTSPEA	0.0270±0.1246	0.0200±0.0070	0.8744±0.3446	1.7359±0.2337
MOPSEA	0.1775±0.3085	1.0768±0.2711	1.4113±0.2081	2.0257±0.2888
RNSGA-II	0.0029±0.3138	1.3054±0.1382	1.4786±0.2097	1.8646±0.1986
MOEARF	0.0134±0.0098	**0.0117±0.0025**	0.0242±0.0087	0.0499±0.0236
Algorithm	DTLZ1(30)-0%	DTLZ1(30)-5%	DTLZ1(30)-10%	DTLZ1(30)-20%
PLREDA	**35.396±58.886**	**140.92±47.737**	**135.14±34.452**	**147.18±35.460**
LREDA	143.91±35.028	206.25±41.229	233.78±48.610	259.58±51.036
REDA	143.91±35.028	222.20±43.111	267.82±45.780	263.98±50.793
NSGA-II	88.702±24.354	810.45±74.198	830.09±82.451	841.14±81.941
NTSPEA	81.192±19.633	266.32±67.300	256.62±59.640	277.40±65.426
MOPSEA	299.93±116.66	413.02±99.206	424.27±115.08	468.61±114.92
RNSGA-II	88.702±24.354	810.45±74.198	830.09±82.451	841.14±81.941
MOEARF	286.56±71.423	535.86±111.55	566.59±91.068	555.32±109.69
Algorithm	DTLZ2(30)-0%	DTLZ2(30)-5%	DTLZ2(30)-10%	DTLZ2(30)-20%
PLREDA	0.0590±0.0075	0.1302±0.0204	0.2008±0.0291	0.3556±0.0446
LREDA	0.0967±0.0129	0.1809±0.0273	0.2469±0.0436	0.3748±0.0672
REDA	0.0967±0.0129	0.1859±0.0325	0.2572±0.0435	0.3802±0.0517
NSGA-II	**0.0549±0.0031**	0.1329±0.0238	0.2178±0.0467	0.4059±0.0807
NTSPEA	0.0679±0.0049	**0.1024±0.0142**	**0.1389±0.0155**	0.3271±0.0811
MOPSEA	0.0741±0.0056	0.1400±0.0153	0.1898±0.0234	**0.2824±0.0338**
RNSGA-II	0.2656±0.0815	0.3343±0.0933	0.3531±0.0869	0.4495±0.0951
MOEARF	0.0829±0.0138	0.2369±0.0505	0.3027±0.0783	0.4318±0.0797
Algorithm	DTLZ3(30)-0%	DTLZ3(30)-5%	DTLZ3(30)-10%	DTLZ3(30)-20%
PLREDA	**79.264±113.35**	**222.60±59.115**	**235.55±62.717**	**218.48±72.925**
LREDA	192.88±43.167	399.92±56.994	388.10±84.239	395.49±79.214
REDA	192.88±43.167	376.40±66.955	396.61±57.952	401.35±80.442
NSGA-II	123.32±34.176	326.11±114.48	279.47±61.408	295.52±72.263
NTSPEA	172.20±88.578	393.38±116.36	365.94±89.696	398.62±118.45
MOPSEA	655.55±227.52	635.81±137.61	678.54±98.491	650.39±132.55
RNSGA-II	822.56±166.19	986.85±107.06	953.77±134.41	988.71±132.33
MOEARF	370.93±77.204	498.70±82.094	511.29±90.213	438.93±162.49

Pareto optimal front. Figure 5.4 shows the Pareto front of DTLZ1 generated from the different algorithms. It could be seen that all of the algorithms failed to converge to global optimality. As a comparison, the results generated from PLREDA were nearer to the global Pareto optimal front. Furthermore, it was observed that REDA, LREDA, and PLREDA have much more non-dominated solutions in their final front than the other five algorithms. Among them, PLREDA had more non-dominated solutions, followed by LREDA and REDA.

5.5.2 Scalability Analysis

In this section, the scalability behaviour of the different algorithms was tested. This is done by spanning the number of decision variables from 30 to 100. Figure 5.5 shows the behaviour of the algorithms in test problem ZDT1, with different number of decision variables measured with

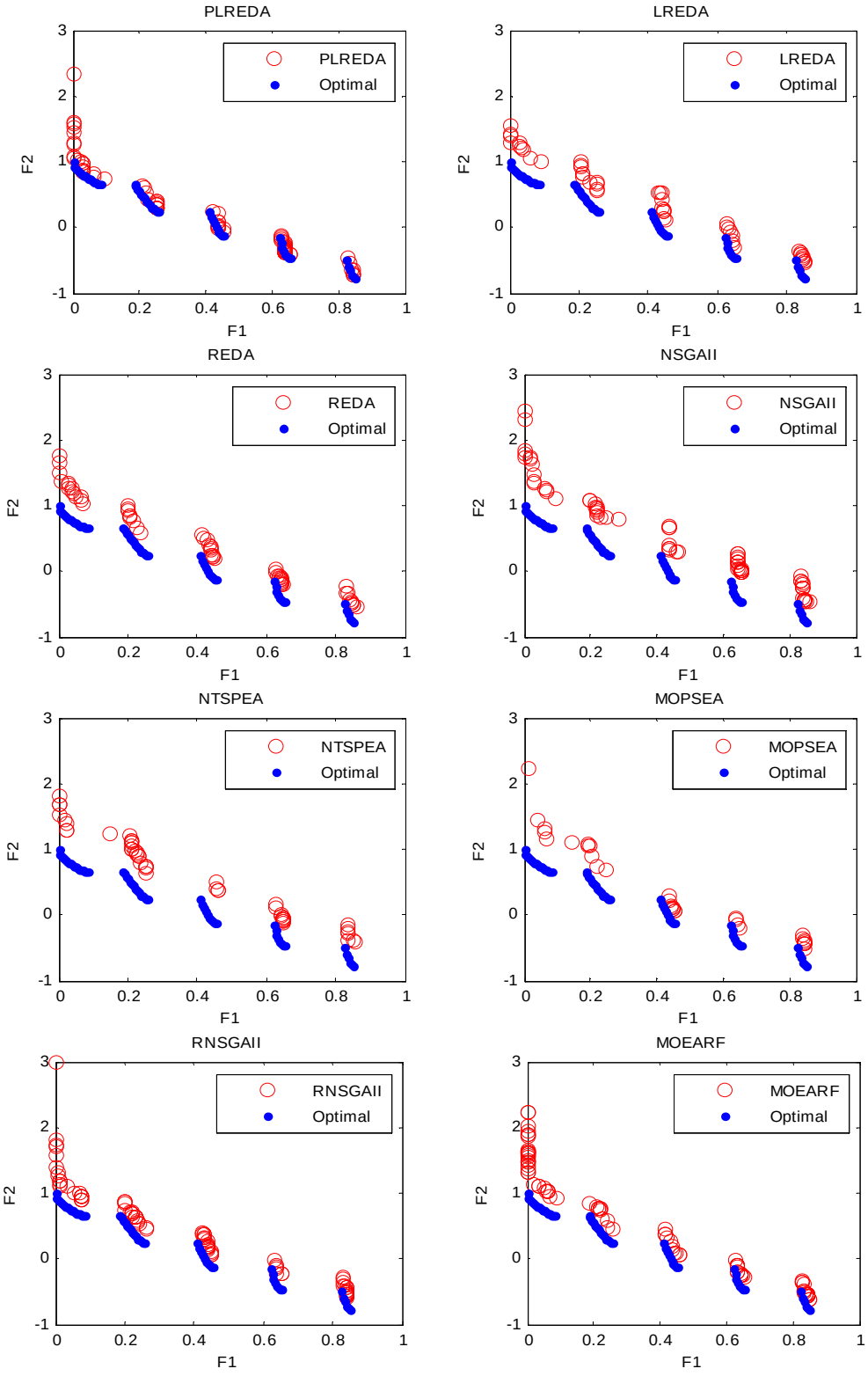


Figure 5.3: Pareto front of ZDT3 generated from the different algorithms

IGD indicator under (a) 0% and (b) 20% noise levels. From Figure 5.5(a), it is observed that the performance of all of the algorithms is decreased when the number of decision variables is

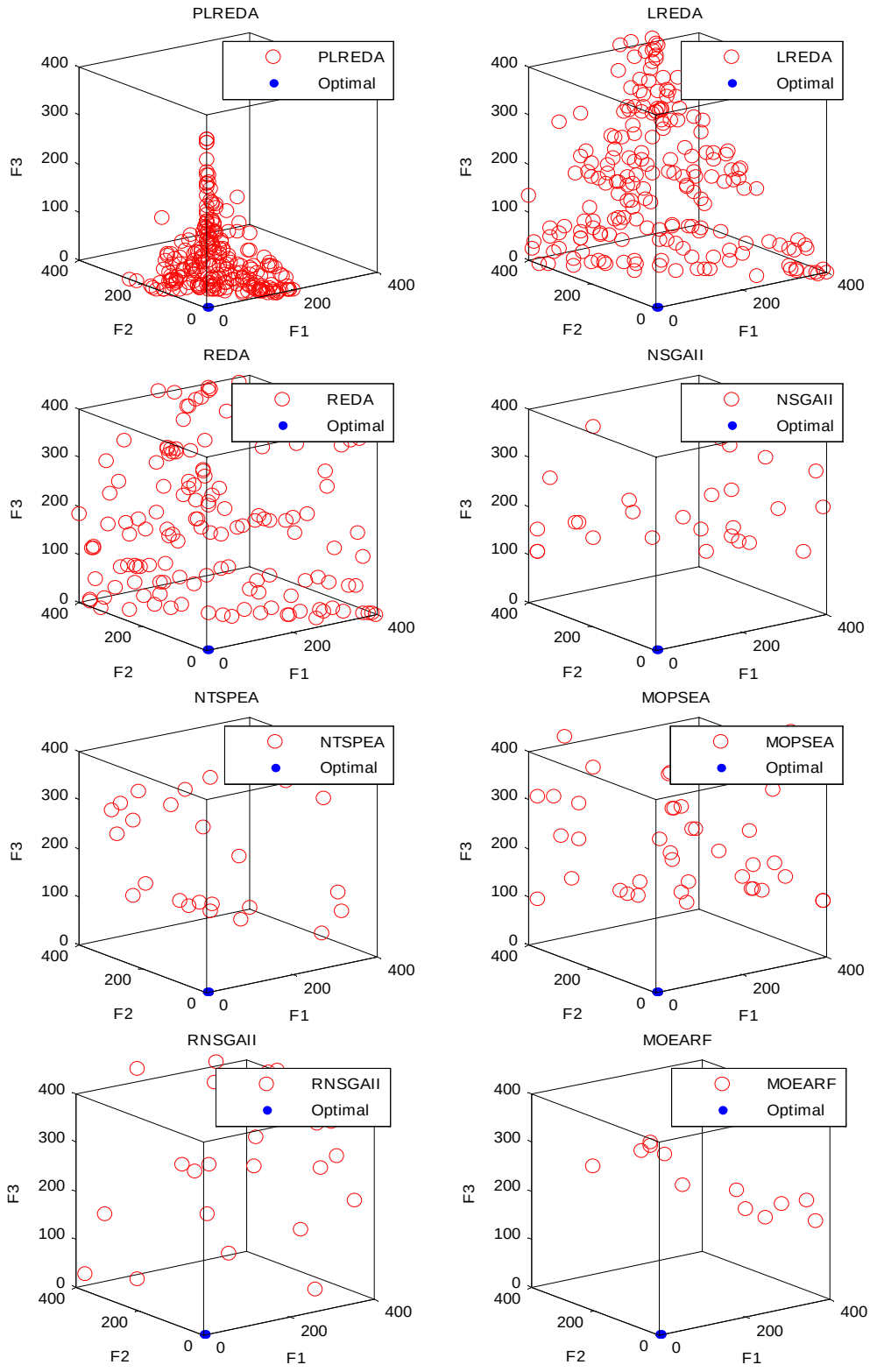
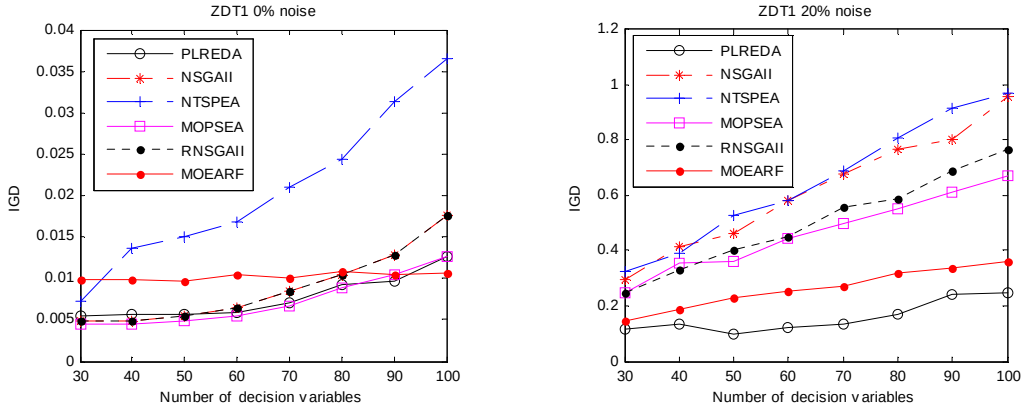


Figure 5.4: Pareto front of DTLZ1 generated from the different algorithms

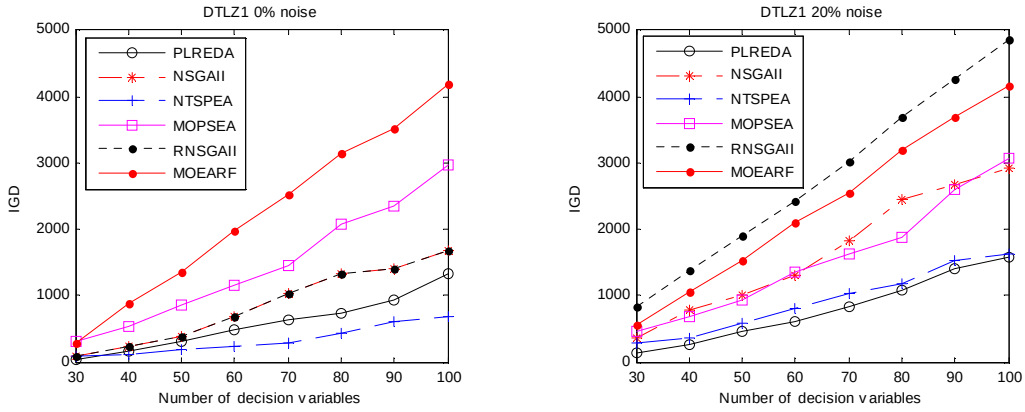
increased. This was expected since the difficulty and complexity of the problem was raised with the increase in the number of decision variables. MOPSEA and PLREDA clearly outperformed

other algorithms. MOEARF is the most robust towards the increase in the number of decision variables as shown by the near-similar IGD values while the performance of NTSPEA was the poorest. In 20% noise level, PLREDA was able to maintain the good performance, thus outperforming the other algorithms. The second best performance was obtained in MOEARF. Other than that, the performance of the other algorithms was very much affected with the increase in the number of decision variables.



(a) (b)

Figure 5.5: Performance metric of IGD versus the number of decision variables in test problem ZDT1 under (a) 0% and (b) 20% noise levels



(a) (b)

Figure 5.6: Performance metric of IGD versus the number of decision variables in test problem DTLZ1 under (a) 0% and (b) 20% noise levels

Figure 5.6 shows the scalability behaviour of the algorithms in test problem DTLZ1 with

different number of decision variables measured with the IGD indicator under (a) 0% and (b) 20% noise levels. The results observed in this problem were quite different from the previous one. In 0% noise level, the performance of NTSPEA was the best compared to other algorithms, while MOEARF showed the worst performance. PLREDA performed better than other algorithms except NTSPEA. Under the influence of 20% noise level, PLREDA outperformed NTSPEA and dominated the overall performance. Even though MOEARF achieved good IGD in ZDT1, its performance was poor in DTLZ1. The worst result was produced by RNSGA-II.

Overall, PLREDA obtained the most promising results in both of the test problems with different number of decision variables. This may be due to the learning capabilities of the RBM. In PLREDA, the probabilistic model was built from the RBM network. The network learnt the multivariate dependencies among the decision variables and returns the information in terms of energy stability. This characteristic has the potential to estimate the global probabilistic distribution to guide the search during the evolution process. In addition, the hybridization with PSO has enhanced the search ability of REDA. This is important as PSO provides a directional search which may explore the promising regions where probabilistic model may fail to explore. In conclusion, PLREDA scaled well with the number of decision variables compared to the other algorithms.

5.5.3 Possibility of Other Hybridizations

In this section, the possibility of other hybridizations with LREDA is investigated. The proposed algorithm, PLREDA, is the hybridization between LREDA and PSO. In order to study the potential of other hybridizations, LREDA is hybridized with a genetic algorithm (GA) and a differential evolution (DE). The simplest and most common GA is applied, where single point crossover and bit-flip mutation are implemented. For DE, the standard recombination proposed in [166] is applied. Table 5.8 shows the IGD values obtained from the different hybridizations. LREDA is the algorithm without any hybridizations. The results are highlighted in bold if the hy-

bridization shows improvement compared to the original algorithm (LREDA). From the charts, it is clear that all hybridizations are able to improve the performance of LREDA, in most of the test problems, under both noiseless and 20% noise level. Among them, hybridization with PSO gave the best results followed by DE and then GA. The function of hybridization is to provide extra search ability for LREDA as LREDA performs the search by using only global statistical information. This hybridization therefore enhances the ability for LREDA in exploring the search space, especially in the early stage of evolutions where the search space is huge. The search using position information (GA, PSO, DE) is also essential and useful especially to explore and exploit certain promising regions. Thus, hybridization is an important mechanism to improve the search performance of EDAs.

Table 5.8: Performance metric of IGD obtained from the different hybridizations

Problem	Noise	PSO	GA	DE	LREDA
ZDT1(30)	0%	0.0054±0.0025	0.0047±0.0003	0.0046±0.0003	0.0042±0.0002
	20%	**0.1178±0.0512**	0.2178±0.0342	**0.1552±0.0206**	0.1960±0.0421
ZDT2(30)	0%	0.0048±0.0004	0.0046±0.0002	0.0046±0.0001	0.0043±0.0001
	20%	**0.1680±0.1202**	**0.3572±0.0751**	**0.3392±0.2217**	0.4973±0.1511
ZDT3(30)	0%	0.0062±0.0030	**0.0055±0.0004**	**0.0054±0.0003**	0.0058±0.0007
	20%	**0.1575±0.0533**	0.2244±0.0178	**0.1790±0.0532**	0.2299±0.0332
ZDT4(10)	0%	**0.3898±0.9901**	**0.4872±0.1955**	**0.4366±0.1898**	0.5403±0.2386
	20%	**0.0581±0.0205**	**0.5409±0.2948**	**0.4369±0.1608**	0.5746±0.2268
ZDT6(10)	0%	0.0027±0.0003	**0.0026±0.0004**	0.0031±0.0005	0.0027±0.0003
	20%	**0.0365±0.0154**	**0.6202±0.5297**	1.3922±0.3021	0.8468±0.5366
DTLZ1(30)	0%	**35.396±58.886**	**89.579±23.450**	**76.113±21.804**	143.91±35.028
	20%	**147.18±35.460**	**202.77±34.360**	**190.10±35.433**	259.58±51.036
DTLZ2(30)	0%	**0.0590±0.0075**	**0.0739±0.0080**	**0.0724±0.0081**	0.0967±0.0129
	20%	**0.3556±0.0446**	0.4113±0.0440	**0.3321±0.0441**	0.3748±0.0672
DTLZ3(30)	0%	**79.264±113.35**	**129.86±23.880**	**123.01±22.637**	192.88±43.167
	20%	**218.48±72.925**	**287.64±79.841**	**244.08±76.056**	395.49±79.214

5.5.4 Computational Time Analysis

Table 5.9: CPU time (s) used by the different algorithms to complete a single simulation run in the different test problems under 0% noise level

Algorithm	Problem(n)		
	ZDT1(30)	ZDT4(10)	DTLZ1(30)
PLREDA	122.51±1.0389	43.499±0.3696	92.536±0.1147
LREDA	231.89±0.7859	75.761±0.3535	176.25±2.1791
REDA	234.14±4.3558	78.111±1.2185	173.56±1.6536
NSGA-II	2.6207±0.1078	1.4713±0.1424	6.0777±0.0685
NTSPEA	11.011±0.5287	9.9386±0.5988	9.7985±1.0396
MOPSEA	12.266±0.8743	12.564±0.6009	46.804±1.1997
RNSGA-II	2.6207±0.1078	1.4713±0.1424	6.0777±0.0685
MOEARF	126.71±7.5289	74.708±6.8302	77.333±6.3065

Table 5.10: CPU time (s) used by the different algorithms to complete a single simulation run in the different test problems under 20% noise level

Algorithm	Problem(n)		
	ZDT1(30)	ZDT4(10)	DTLZ1(30)
PLREDA	120.70±0.8073	43.114±0.2843	91.133±0.4819
LREDA	222.81±0.4885	76.617±0.1805	175.95±0.5336
REDA	228.62±8.8671	76.326±0.2052	172.70±1.0873
NSGA-II	2.1863±0.0748	0.9991±0.1401	5.7052±0.1372
NTSPEA	1.7295±0.1511	0.8452±0.1021	12.662±0.5601
MOPSEA	4.0765±0.1056	3.8687±0.1763	65.694±1.3063
RNSGA-II	0.7892±0.1476	0.4004±0.0485	2.0300±0.0192
MOEARF	23.493±0.2349	4.0263±0.1019	93.594±6.8869

The incorporation of any noise handling features or enhancement operators will incur additional computational cost. In this section, the CPU time of the different algorithms are studied. Table 5.9 shows the CPU time (s) used by the different algorithms to complete a single simulation run under 0% noise level. It was observed that REDA and LREDA were the most time consuming algorithms, followed by PLREDA and MOEARF. The rest of the algorithms took less computational times. The CPU time (s) used by the different algorithms to complete a single simulation run under 20% noise level is tabulated in Table 5.10. The CPU times used by REDA, LREDA, PLREDA, and NSGA-II in solving test problems with two objective functions were not very different from that in the noiseless circumstance, while there was much less computation needed for the other algorithms. REDA was the most time consuming algorithm. This is due to the training and modelling part in RBM. In REDA, training is conducted at each generation and stops when the number of training epochs is reached. This training process is more complicated than the genetic operators in the other MOEAs, thus, incurring extra computational time. The incorporation of likelihood correction into REDA increased the CPU time of LREDA slightly, however, the extra time incurred was less than the training time in REDA. As for PLREDA, the computation of PSO was fast. Therefore, PLREDA took less CPU time than LREDA and REDA. The noise handling mechanism in NTSPEA, MOPSEA, and MOEARF measured certain information (e.g. domination behaviour among all solutions) in order to cope with noisy information. In noisy conditions, the number of non-dominated solutions was fewer than in noiseless situations. Thus, NTSPEA, MOPSEA, and MOEARF took less computational time in noisy environments than in

noiseless circumstances. Even though PLREDA took more CPU time than other MOEAs with genetic operators, the time taken to perform a single simulation run was acceptable. This is because most of the real-world optimization cost functions are very time consuming, for example a few minutes is required for a single fitness evaluation.

5.6 Summary

This research studied the potential of an MOEDA in dealing with noisy MOPs. REDA has been used to tackle the noisy information. The presence of noise in the objective functions may cause weaker individuals to be selected to the next generation. Therefore, a likelihood correction scheme, which utilizes the information of probability error in selection, was proposed. This feature adjusts the independent marginal distribution in each decision variable according to the heuristic that the true probability distribution of the solutions is more likely to follow the distribution of the individuals with smaller probability selection errors. The REDA, which uses only global information in guiding the search, has limitations in exploring certain promising search spaces. In order to improve the search ability, a particle swarm optimization algorithm is hybridized with REDA. The empirical results showed that the proposed algorithm outperformed most of the algorithms in most of the test instances and has better scalability. Finally, the hybridization between EDAs and other evolutionary paradigms, particularly GA and DE, has been shown to improve the search performances.

Chapter 6

Application of REDA in Solving the Travelling Salesman Problem

This chapter presents the application of evolutionary algorithms for bi-objective travelling salesman problem. Two evolutionary algorithms, including estimation of distribution algorithms (EDAs) and genetic algorithms (GAs), are considered. The problem is solved by optimizing the order of the cities so as to simultaneously minimize the two objectives of travelling distance and travelling cost incurred by the travelling salesman. In this chapter, the binary-number representation-based evolutionary algorithms are replaced with an integer-number representation. Three existing EDAs are altered to use this integer-number representation, namely restricted Boltzmann machine (RBM), univariate marginal distribution algorithm (UMDA), and population-based incremental learning (PBIL). Each city is associated with a representative integer, and the probability of any of this representative integer to be located in any positions of the chromosome is constructed through the modelling approach of the EDAs. New sequences of cities are obtained by sampling from the probabilistic model. A refinement operator and a local search operator are proposed in this piece of work. The EDAs are subsequently hybridized with a GA in order to complement the limitations of both algorithms. The effect that each of these operators has on the quality of the solutions are investigated. Empirical results show that the

hybrid algorithms are capable of finding a set of good tradeoff solutions.

6.1 Introduction

The travelling salesman problem (TSP) is a famous permutation-based combinatorial optimization problem which has been extensively studied over the past few decades. Despite its simplicity in understanding and general applicability to most scheduling problems, the TSP is notoriously difficult to solve. Researchers have recognized it as an NP-hard problem [7, 167]. Some problems which can be described using the TSP formulation are the gene sequencing problem [168], the dartboard design problem [169], the hole punching problem [170], etc. Additional literature review of the development of TSP can be found in [171, 172].

The TSP aims to minimize the total distance travelled, in which each city can only be visited once and the salesman must return to the starting depot after visiting the ordered cities. The coordinate of the cities are known in advance in order to find the pairwise distance of the cities. The adaptation of the TSP into multi-objective framework (MOTSP) is a promising area which can be further explored [173]. In MOTSP, the aim is to simultaneously optimize several conflicting objectives, such as shortest travelling distance, minimum time, minimum cost, and lowest risk [174]. In the context of multi-objective, no single point is considered as an optimal solution, because an improvement in one objective will cause at least another objective not being able to be optimized. Therefore, the optimal solution can only be a set of non-dominated and tradeoff solutions.

Many practical problems are closely related to the MOTSP, such as logistic, planning, transportation, and gene sequencing problems. One recent application of the MOTSP is the study of berth allocation problem in container ports [175]. The problem aims to find the optimal berth schedule which simultaneously minimizes the three objectives of waiting time, makespan, and degree of deviation from a predetermined schedule. A fixed-length chromosome representation was proposed which represents a complete berth schedule. In a chromosome, the order of the

ships that are allocated to the berth is determined by the genetic operators. In bioinformatics field, the concept of MOTSP is slightly modified and used to solve the deoxyribonucleic acid (DNA) sequencing problems [176]. Given the spectrums that are extracted from DNA-related experiments, such as DNA microarrays, the problem aims to find a complete sequence of DNA. In this problem, the cities are replaced by the DNA spectrums and the distance between the cities is replaced by the criterion like a similarity measure between the spectrums.

Over the past few decades, various approaches have been proposed to solve the TSP. They include linear programming [177], dynamic programming [178], branch and cut algorithm [179], simulated annealing [180], tabu search [181], evolutionary algorithm (EA) [182], neural network [183], clustering [184], etc. Among them, the use of EAs to solve the TSP and MOTSP has attracted considerable attention [185–188]. This may be attributed to the fact that EAs, which are probabilistic-based algorithms, are less susceptible to be trapped at local optima. In addition, the population-based approach in EAs has the capability to generate a set of tradeoff solutions in just a single simulation run. Selection and reproduction operators inspired from biological evolution enable the algorithms to deal with hard problems which of high dimensionality and having a large search space.

Genetic algorithms (GAs) are famous computing paradigms inspired by biological evolution. Over the past few decades, there were several remarkable research efforts to drive GAs to handle the TSP. However, it is commonly known that the stochastic recombination operators of GAs may disrupt the building of good schemas [7]. An alternative to the use of GAs is the estimation of distribution algorithms (EDAs). EDAs are motivated by the concept of using probability distribution of the candidate solutions to predict the movement in the search space. Another important feature of EDAs is the exploitation of the linkage information between the decision variables which will be used to guide the search process [7, 20, 126, 189, 190].

Solving scheduling problems (specifically the TSP) with the use of EDAs is a new area of research that has been explored recently. Over a past decade, several attempts have been devoted

to studying the potential of EDAs in solving single-objective permutation-based problems. However, there have no research which studies multi-objective permutation-based problems (specifically the MOTSP) using EDAs. In this chapter, three MOEDAs in the binary-number representation, namely multi-objective univariate marginal probability algorithm (MOUMDA) [20], multi-objective population based incremental learning (MOPBIL) [191], and multi-objective restricted Boltzmann machine (REDA), are being adapted into a permutation-based integer-number representation to solve the bi-objective TSP. These algorithms utilize the principles of Pareto optimality to deal with the multiple conflicting objectives. A permutation refinement operator is proposed to refine the cities in a chromosome to guarantee that no city is repeated. A local search operator is also presented to enhance the search capability of the algorithms. EDAs are subsequently hybridized with a GA to improve the diversity of the tradeoff solutions. The influences of each operator and parameter settings are also studied.

The rest of this chapter is organized as follows. Section 6.2 describes the formulation of the travelling salesman problem in a multi-objective framework, and an overview of existing studies. Section 6.3 presents the algorithmic framework, together with their corresponding operators. Section 6.4 outlines the implementation settings of this study. The experimental results, investigations, and discussions are presented in Section 6.5. This chapter ends off with the conclusion in the last section.

6.2 Background Information

In this section, the mathematical formulation of the MOTSP and an overview of the relevant studies are presented.

6.2.1 Problem Formulation

The TSP is a combinatorial optimization problem which is applicable to many scheduling problems. The aim of the problem is to find the route that has shortest travelling distance or lowest

travelling cost for visiting all the cities. Each city is only can be visited once and the salesman must return to the starting depot after visiting the ordered cities. Mathematically, the problem in multi-objective framework [192] can be formulated, for the minimization case, as follows:

Minimize:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x})) \tag{6.1}$$

where

$$f_1(\mathbf{x}) = \sum_{i=1}^{n-1} D^1(x_i, x_{i+1}) + D^1(x_n, x_1)$$

$$\vdots$$

$$f_m(\mathbf{x}) = \sum_{i=1}^{n-1} D^m(x_i, x_{i+1}) + D^m(x_n, x_1)$$

where m is the number of objective functions, n is the number of cities, x_i is the representative integer of the city at i^{th} position of a chromosome, $D^m(x_i, x_{i+1})$ can be considered as the travelling distance, travelling cost, or travelling risk (for the m^{th} objective function) between cities at i^{th} and $(i+1)^{th}$ positions of a chromosome. The constraint in the problem is that all the cities must be visited exactly once in a route. In other words, there should be no repeated visit on any cities in the route of the salesman. Readers can refer to [192–195] for a more detailed study of the MOTSP. In this chapter, only two objectives (say, travelling distance and travelling cost) are considered.

6.2.2 Existing Studies

Over the past decade, several attempts have been devoted to implementing EDAs for solving single-objective permutation-based problems. In [7], the first adaptation of EDAs to solve the TSP was carried out. Several EDAs with discrete-number and continuous-number representa-

tions were utilized. Domain knowledge was used as a way to introduce a local search operator to facilitate the search. In [196], the guidelines of implementing EDAs in scheduling problems were presented. The authors showed that the intensification and diversification effects of EDAs can be enhanced by hybridizing EDAs with other meta-heuristic approaches. Next, Jarboui *et al.* [197] studied the permutation flowshop scheduling problem using UMDA [20]. In the implementation, the order of the jobs to be performed in the sequences and the similar blocks of the jobs were modelled. A variable neighbourhood search operator, which is used to determine when and how the local search is to be performed, was also proposed to enhance the search. In [198], the authors studied the hybrid flow shop with sequence dependent setup times and uniform machines in parallel. The population-based incremental learning (PBIL) [191] was used, and the learning rate in PBIL was investigated. A guided mutation strategy was also adapted in PBIL to enhance the exploitation capability of the algorithm. In [199], a new EDA based on maximum entropy was proposed to deal with the job shop scheduling problem. In [200], EDA was used to estimate the distribution of the state transitions and the global updating rules of an ant colony system. In this case, EDA is not applied directly to deal with the TSP; but acted as an optimization tool to update the parameters in the ant colony system. Most of the mentioned studies carried out the performance comparison between EDAs and GAs, and the experimental results indicated that the performance of EDAs, when they are hybridized with local search algorithms, is better than GAs in terms of solution quality and convergence speed.

6.3 Proposed Algorithms

In this section, the overall framework of the proposed algorithms and the corresponding operators are being presented. Firstly, the integer-number representation of a chromosome will be described. Then, the modelling and sampling approaches of EDAs and recombination of GAs are presented. In the original version of EDAs, the binary-number representation is used. However, since integer-number representation is used in this study, the modelling and sampling approaches

are modified accordingly. Subsequently, the description of the enhanced operators, namely permutation refinement operator and local search operator, are presented. The overall framework of the algorithms is outlined in the final sub-section.

6.3.1 Permutation-based Representation

The chromosome is represented by a set of sequenced cities, as shown in Figure 6.1. Each chromosome (ch1 and ch2) encodes a complete solution. Integer value is used to represent the cities, where 1 means the first city. The gene of a chromosome will therefore consist of different integer values ranging from 1 to n (where n is the maximum number of city to be visited). The sequence of the cities to be visited is denoted by the sequence which it appears in the chromosome.

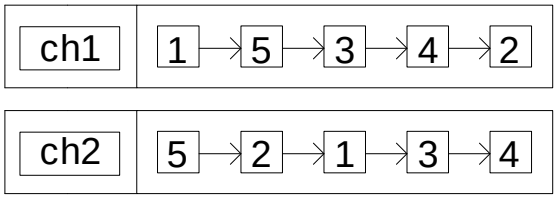


Figure 6.1: Integer-number representation

6.3.2 Fitness Assignment

Pareto-based ranking and crowding distance is one of the favorite fitness assignment operators, thus, it is implemented here. The detail operation of the Pareto-based ranking and crowding distance have been described in Section 2.3.1. After assigning a fitness value to the archived solutions, binary tournament selection is used to select the archive solutions to reproduce in next generation. Under this selection scheme, two chromosomes are randomly selected to undergo tournament selection by comparing their rank of domination. The solution with a lower rank is preferred, and thus it will survive to the next generation. However, if two solutions are located at the same rank, then the one that has a higher crowding distance is preferred.

6.3.3 Modelling and Reproduction

This section describes the three different types of EDAs as well as a genetic algorithm (GA). Three EDAs, including univariate marginal distribution algorithm (UMDA), population-based incremental learning (PBIL), and restricted Boltzmann machine (RBM), are considered in this chapter. The original versions of the EDAs are in binary-number representation. In order to deal with the MOTSP, the probabilistic modelling and sampling approaches are modified to handle the integer-number representation.

Univariate marginal distribution algorithm (UMDA)

UMDA [91] uses univariate modelling to construct the probability distribution of the cities in each position of the chromosome without considering the linkage dependencies between the cities. In the modelling, a $n \times n$ probability matrix, which models the probability of the cities, is constructed as follows:

$$p_g(\mathbf{x}) = \begin{bmatrix} p_g(x_1 = c_1) & \dots & p_g(x_n = c_1) \\ \vdots & \ddots & \vdots \\ p_g(x_1 = c_n) & \dots & p_g(x_n = c_n) \end{bmatrix} \tag{6.2}$$

where $p_g(\mathbf{x})$ is the probability distribution of the cities at generation g, $p_g(x_i = c_j)$ is the probability of city j to be located at the i^{th} position of the chromosome, c_j is the city j ($c_j = j$), and n is the number of cities. This modelling considers the frequency of existence of the cities in each location of the chromosome. The probability of the cities in each position of the chromosome is calculated as follows:

$$p_g(x_i = c_j) = \frac{\sum_{k=1}^{N} \delta_k(x_i = c_j) + \frac{1}{n}}{N + \frac{N}{n}} \tag{6.3}$$

$$\delta_k(x_i = c_j) = \begin{cases} 1 & \text{if } x_i = c_j \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

where N is the population size. The terms $\frac{1}{n}$ and $\frac{N}{n}$ are added to set the upper and lower bounds to the probability of each city. This is important as the probability of 0.0 and 1.0 will make no progress in future evolutions since a probability of 0.0 means that there will never be a generation of this particular city in the position of the chromosome. Similarly, a probability of 1.0 will mean that there will always be the generation the same city in the same position of the chromosome.

Population-based incremental learning (PBIL)

PBIL [191] is another version of EDAs which uses the same modelling approach as UMDA. The primary difference between the PBIL and the UMDA is in terms of its probability updating rule. In PBIL, the probability of the cities in each position of the chromosome is calculated by considering the probability of the cities in current and previous generations. The updating rule is presented below:

$$p_g^{'}(x_i = c_j) = \alpha p_g(x_i = c_j) + (1 - \alpha) p_{g-1}(x_i = c_j) \tag{6.5}$$

where $\alpha \in [0, 1)$ is the learning parameter of the algorithm, $p_g^{'}(x_i = c_j)$ is the final probability of city j at the i^{th} position of the chromosome at generation g, $p_g(x_i = c_j)$ is obtained according to equation (6.3). α is set to 1.0 at the first generation since there is no prior probability distribution from any previous generations. In this situation, PBIL is similar to UMDA. In order to differentiate PBIL from UMDA, α would never be set to 1.0 over course of the evolution process.

REDA

The modelling and sampling mechanism of REDA in the binary-number representation have been discussed in Chapter 3. In the integer-number representation, the probabilistic model of the representative solutions is constructed in the following equations:

$$p_g(x_i = c_j) = \frac{\sum_{k=1}^{N} \delta_k(x_i = c_j) + \frac{Z_i}{N \times n}}{Z_i + \frac{Z_i}{N}} \tag{6.6}$$

$$\delta_k(x_i = c_j) = \begin{cases} \sum_{l=1}^{H} e^{-E(v_i^k = c_j, h_l)} & \text{if } x_i = c_j \\ 0 & \text{otherwise} \end{cases} \tag{6.7}$$

$$Z_i = \sum_{k=1}^{N} \left(\sum_{l=1}^{H} e^{-E(x_i^k, h_l)} \right) \tag{6.8}$$

where v is the input state, h is the hidden state, Z is the normalizing constant, E is the energy value of the network as presented in equation (3.1), and N is the population size. The binary-number representation is used in the original REDA. Since the integer-number representation is utilized in this implementation, a decoding scheme from integer number to binary number is needed. 7 bits binary number is used to decode 100 cities, 8 for 200 cities, and 10 for 500 cities. It is to be noted that 9 bits binary number is able to decode 500 cities. In the experiments, we arbitrarily choose 10 bits instead of 9 bits. In fact, any number of bits can be used as long as the possible states of the binary bits are enough to represent the number of cities. For example, 7 bits have 128 possible states, thus, is possible to decode 100 cities. After training, an encoding scheme is performed to obtain the probability of the cities in each position of the chromosome. Figure 6.2 shows the architecture of RBM and it adaptation in the integer-number representation. The overall modelling procedure is summarized in Figure 6.3.

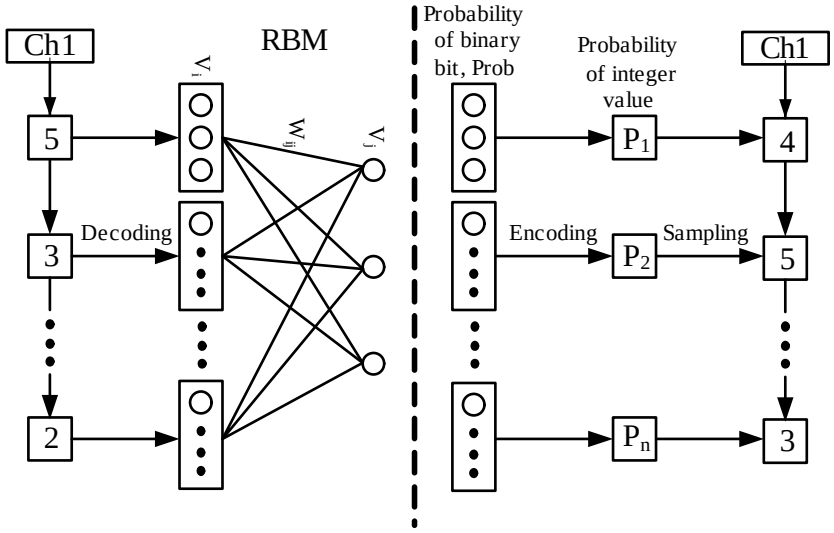


Figure 6.2: RBM framework in the integer-number representation

Begin

1. Render the candidate solutions as input vectors into the visible units of an RBM
2. Decode the integer-number representation of the cities into the binary-number representation
3. Train the network using CD training mechanism to obtain the trained weights and biases
4. Compute the energy value of the solutions
5. Compute the probability of the cities in binary-number representation
6. Encode the binary-number representation into integer-number representation. Obtain the final probability distribution of the cities by taking average of the probability computed in Step 5

End

Figure 6.3: Probabilistic modelling using RBM in the integer-number representation

Sampling

All EDAs considered in this chapter will output the probability distribution in a similar form, which is the probability distribution of the cities to be located in each position of the chromosome. Therefore, the same sampling procedure can be applied to all EDAs. Offspring are generated by sampling the computed probability distribution, according to the following equation:

$$y_j = \begin{cases} c_1 & \text{if random } (0,1) \leq p_g\left(x_j = c_1\right) \\ c_2 & \text{if } p_g(x_j = c_1) < \text{random } (0,1) \leq \sum_{i=1}^{2} p_g(x_j = c_i)) \\ & \vdots \\ c_n & \text{if } \sum_{i=1}^{n-1} p_g(x_j = c_i) < \text{random}(0,1) \leq \sum_{i=1}^{n} p_g(x_j = c_i) \end{cases} \tag{6.9}$$

where y_j is a newly generated city at j^{th} position of a chromosome, random (0,1) is a randomly generated value between [0, 1], and C_n is the city n.

Genetic Algorithm (GA)

In a genetic algorithm (GA) [1], its variation operators are crossover and mutation. Single-point crossover is used to create offspring. This operator randomly selects the position to cut the chromosomes for crossing over between two parents. This single-point crossover is equivalent to the route inter-crossing. Through this biological process, the offspring will inherit some properties of the parent solutions. After which, a mutation strategy is performed by swapping between two randomly selected alleles within a chromosome. This genetic perturbation provides exploitation ability to the optimizer to search within fitter regions.

6.3.4 Feasibility Correction

One of the problems after reproduction is that some cities may not be visited at all, while others are visited more than once. In EDAs, the sampling mechanism does not consider which city has or has not been included in the route. The same problem occurs in the GA, where the recombination of the alleles between two parents is done indiscriminately. To overcome this problem, the simple permutation correction can be used. First of all, this correction identifies the repeated cities and the unvisited cities. The repeated cities are subsequently deleted and randomly replaced by the unvisited cities. However, this methodology is only able to satisfy the constraints of the problem, but it does not provide any extra information to guide the search. To

fully utilize the available information from the database (travelling distance and travelling cost between cities), a refinement operator is proposed. Firstly, the repeated cities in a chromosome are detected, which will be followed by the recording of the unvisited cities. In this case, the number of repeated and unvisited cities should be the same. An insertion step is then carried out by inserting the unvisited cities to the position of the repeated cities. The average distance and cost (Score1 as depicted in step 3 of Figure 6.4) between the adjacent cities in the permutation are calculated and they will serve as the main criterion for insertion. For example, if the first unvisited city has a smaller average value of the travelling distance and cost compared to those of the second unvisited city at the first position of the repeated cities, then this city is inserted to the first position and the second unvisited city is inserted to another position. The pseudo-code of the refinement operator is presented in Figure 6.4.

Begin
For $p = 1: N$
1. Identify the location of the repeated cities in chromosome p, $L_i = L_1, \dots, L_R$
2. Identify the unvisited cities in chromosome p, $V_j = V_1, \dots, V_R$
For $i = 1: R$
3. Calculate the score for all unvisited cities V_j in location L_i according to the following equation:

$$Score1\ (j) = \frac{\sum_{q=1}^{m} D^q\left(V_j,\ x_{L_i+1}\right) + D^q\left(V_j,\ x_{L_i-1}\right)}{m}$$

4. Insert a city in V_j which has the smallest score, to the $L_i{}^{th}$ position of the chromosome. Discard that city from V_j
End For i
End For p
End

Figure 6.4: Pseudo-code of the refinement operator

6.3.5 Heuristic Local Search Operator

In the literature, many studies have been carried out to prove the advantages of using local search operator to enhance the exploitation capability of EAs [7, 51]. Their attempts rely largely on

Begin
1. Pre-define the number of cities to be relocated, k
For $\boldsymbol{p = 1: N}$
%If local search is performed
2. Start at the position ℓ (randomly determined) of a chromosome, and then select a sequence of k cities U_j, where $\ell + k - 1 \leq n$
For $\boldsymbol{q = 1: k}$
3. Calculate the score for all cities in the selected sequence, $U_j = U_1, \ldots, U_k$

$$Score2\ (j) = \frac{\sum_{h=1}^{m} D^h\left(U_j,\ x_{\ell-2+q}\right)}{m}$$

4. Insert a city in U_j which has the smallest score to the $(\ell - 1 + q)^{th}$ position of the chromosome. Discard that city from U_j
End For $\boldsymbol{q}$
End For $\boldsymbol{p}$
End

Figure 6.5: Pseudo-code of the local search operator

the available knowledge of the database (distance, cost, time, etc). With n cities, the TSP will have $(n-1)!/2$ routes. The huge search space of a TSP poses a challenge for optimizers in finding the shortest distance. However, the difficulty of the problem is alleviated when the domain knowledge is taken into consideration during optimization process. In this implementation, a local search operator is proposed. It makes use of the shortest distance and lowest cost among the cities. The process flow of this operator is as follows. Firstly, the number of cities to be relocated (k) is pre-defined. Then, a position l is randomly determined. From this position, a sequence of k cities is being selected. A score based on the distances and costs (Score2 as depicted in step 3 of Figure 6.5) among all the selected cities are then calculated. The permutation of the cities is re-determined according to the Score2. The pseudo-code of the local search operator is presented in Figure 6.5.

6.3.6 Algorithmic Framework

The algorithmic process flow of the proposed algorithm is shown in Figure 6.6. Firstly, the optimizers read the database which contains the coordinates of the cities and the travelling cost between the cities. A $n \times n$ distance matrix from city i to city j is constructed by computing

the Euclidean distance between the cities. Subsequently, population initialization is performed by generating the permutations of the cities randomly. All solutions in the population are evaluated according to equation (6.1) to obtain the objective values. Fitness is then assigned to all the solutions in the population based on the Pareto-based ranking and crowding distance. Next, the binary tournament selection is applied to determine a set of N promising candidate solutions. In the selection process, two chromosomes are randomly picked into tournament. A fitter solution in terms of the lowest rank or highest crowding distance is selected. This selection procedure is repeated until the population size is reached. Subsequently, the probabilistic model of the candidate solution set is built. The probabilistic model is constructed using equations (6.3), (6.5), or (6.6) according to which algorithm is performed. Based on the constructed model, sampling is carried out to produce N offspring according to equation (6.9). For the GA, the single-point crossover and swapping mutation strategy are performed instead of probabilistic modelling and sampling to produce offspring. After reproduction, some cities may be visited more than once while other are never visited. Therefore, the refinement operator is implemented to correct the reproductive law. After this process, the chromosomes should satisfy the constraint of the MOTSP, where each city is strictly visited once. To further improve the routing, a local search operator is incorporated. The local search will only be performed if the generated random value is smaller than a predefined local search rate LS. Subsequently, all offspring are evaluated to obtain their objective values. After this, an archive is created to store the parent and offspring found during the evolutionary process. The solutions in the archive are ranked and the crowding distance of all solutions is computed. Following the ranking criteria, N solutions with the lowest rank or highest crowding distance are chosen. This is an elitism mechanism which may enhance the convergence speed and generate better solutions. This marks the completion of one generation. The same procedure is iterated over generations until the stopping criterion is met. The algorithmic flow of the GA is similar to EDAs, except that the modelling and sampling mechanisms in EDAs are replaced by the crossover and mutation operators. For hybrid algorithms, they start with an

EDA and alternates with the GA every 500 generations.

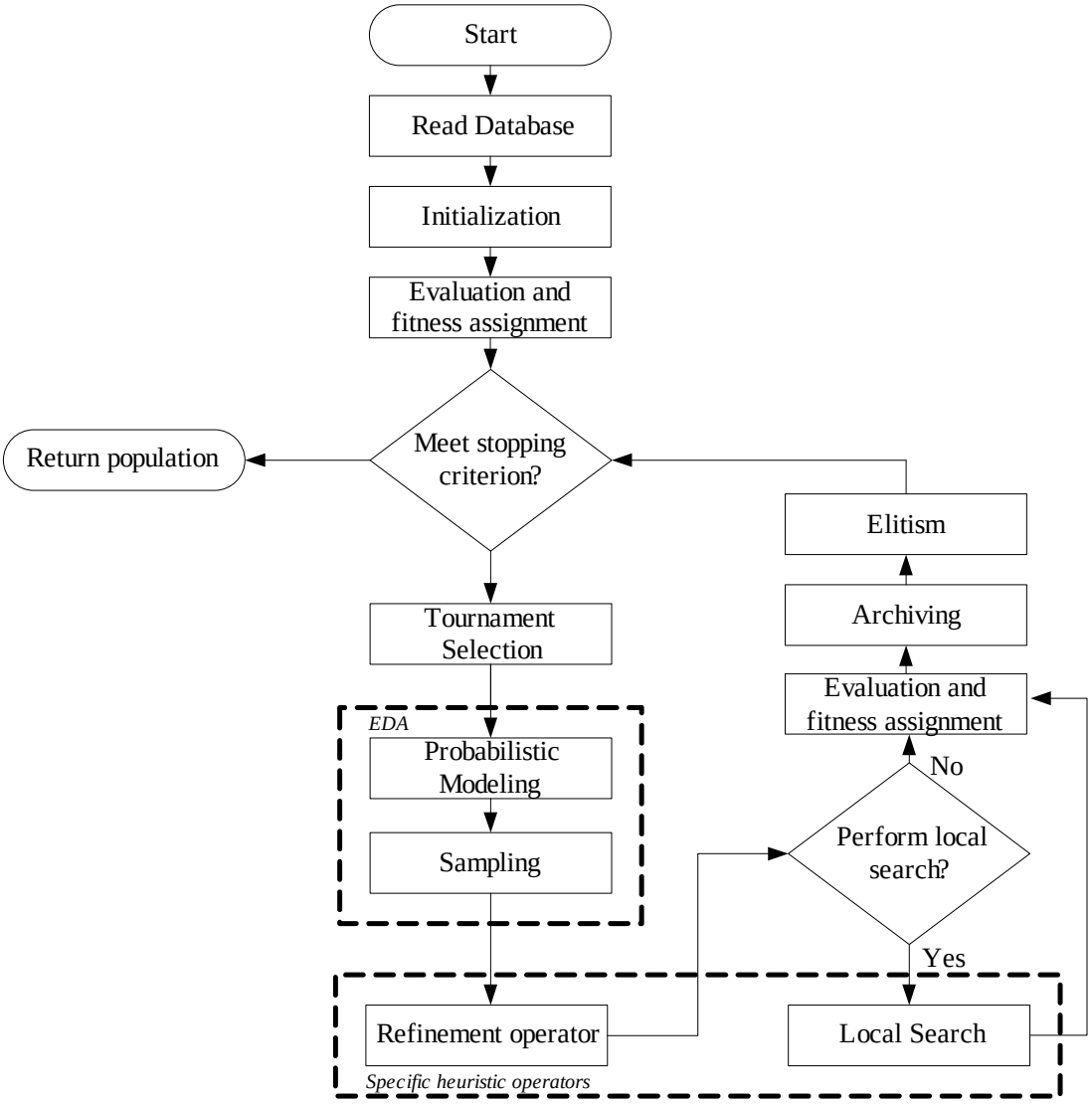


Figure 6.6: Process flow of the MOEDAs

6.4 Implementation

All algorithms were written in C++ and the experimental studies were ran on an Intel(R) Core(TM)2 Duo CPU, 3.0 GHz. The experimental settings are shown in Table 6.1. In this work, the MOTSP with two objectives is studied. The coordinate of the cities and the travelling cost between the cities are randomly generated for each objective in the range of [0, 1000] as done in Peng *et*

al. [192].

Table 6.1: Parameter settings for experiments

Parameter	Setting
Population size, N	Number of cities
Archive size, $2N$	$2 \times$ Number of cities
Number of cities, n	100, 200, 500
Fitness evaluations	$2000N$
Local search rate, Ls	0.5
Frequency of alternation, fr	500
Crossover rate in GA	0.8
Mutation rate in GA	0.05
Independent runs	10
α in PBIL	0.9
Hidden unit in RBM, H	10
Training epoch in RBM	10
k	10

For experimental studies, the results are compared based on the performance metrics of inverter generational distance (IGD) [201], generational distance (GD) [202], maximum spread (MS) [101], and non-dominance ratio (NR) [39]. A smaller IGD value implies better performance in terms of closer proximity of the evolvable front to the approximate Pareto optimal front and a wider distribution of the evolved front along the approximate Pareto optimal front. A smaller GD value shows that the evolvable front is closer to the approximate Pareto optimal front. On the other hand, a higher value of MS means that the evolvable front has better coverage along the approximate Pareto optimal front. Since the optimal solution set to the problem is unknown, the approximate optimal solution set is formed using the non-dominated solutions found from all algorithms and all experimental runs, as has been done in [192]. Lastly, a higher value of NR means that the algorithm produces more non-dominated solutions.

In this section, the empirical comparison of the EDAs, GA, and hybrid algorithms are carried out. For the hybrid algorithm between the EDA and GA, it is start off with EDAs, and each algorithm is being alternated every 500 generations. Next, the effect of the permutation refinement operator, local search operator, and number of alternations in the hybrid algorithms are investigated. Lastly, computation time analysis is presented. In total, seven algorithms are put into comparison. For easier readability of the chapter, the abbreviations of the algorithms are illustrated in Table 6.2.

Table 6.2: Algorithms' abbreviation

Algorithm's abbreviation	Description
RBM-GA	A hybrid algorithm between RBM and GA
UM-GA	A hybrid algorithm between UMDA and GA
PB-GA	A hybrid algorithm between PBIL and GA
RBM	Restricted Boltzmann Machine-based EDA
UMDA	Univariate Marginal Distribution Algorithm
PBIL	Population-based Incremental Learning
GA	Genetic Algorithm

6.5 Results and Discussions

6.5.1 Comparison Results

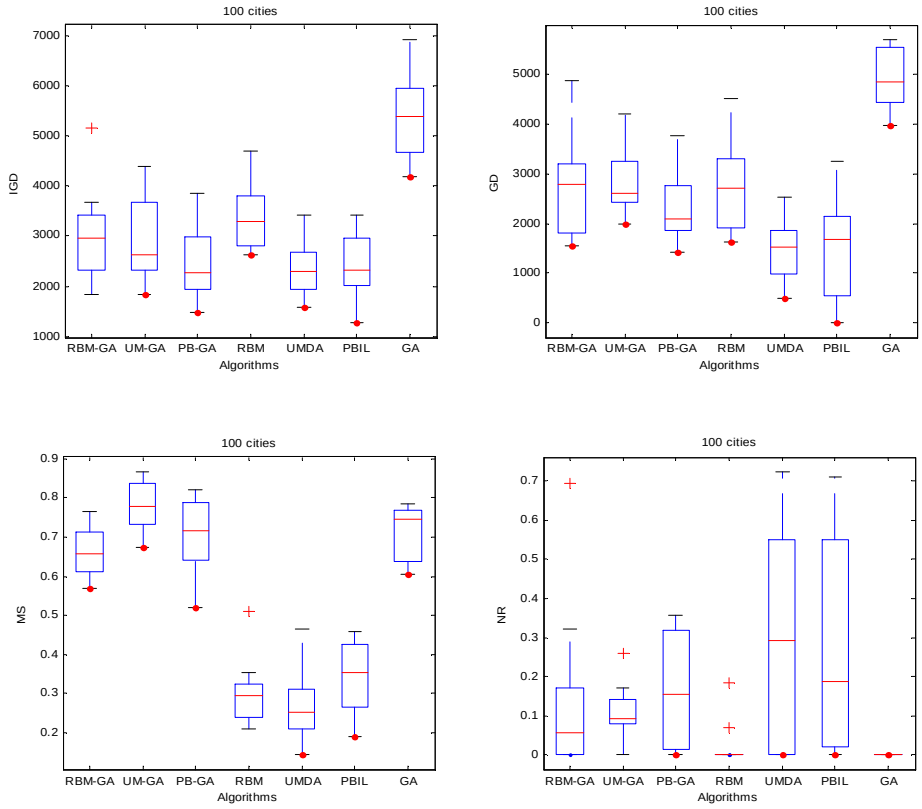


Figure 6.7: Performance metric of IGD, GD, MS, and NR after 200,000 fitness evaluations for MOTSP with 100 cities. X-axis is the algorithms and y-axis is the value of the performance indicators.

Figure 6.7 shows the performance metric of IGD, GD, MS, and NR after 200,000 fitness evaluations for MOTSP with 100 cities. From the IGD performance indicator, it is observed that EDAs (RBM, UMDA, and PBIL) outperform GA. Among the different EDAs, the performance of UMDA and PBIL is better than RBM. When hybridization is carried out, the performance of RBM (RBM-GA) is improved. However, no significant improvement is observed in UM-GA and

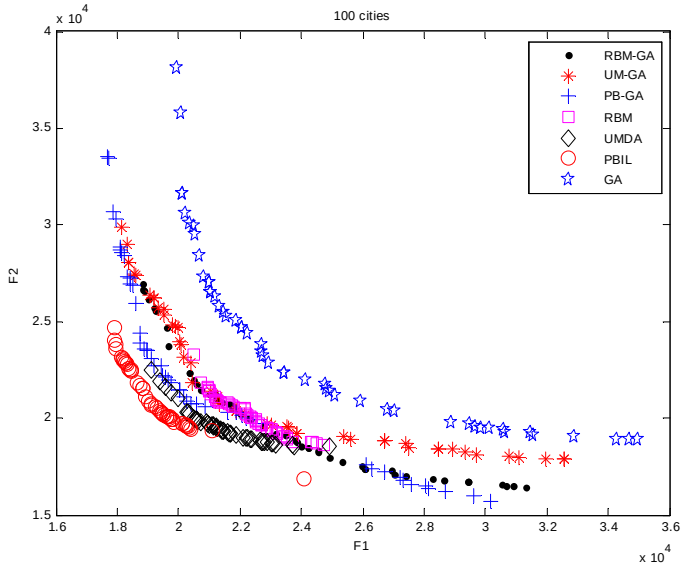


Figure 6.8: Final evolvable front generated by the various algorithms for MOTSP with 100 cities. F1 is the first objective or travelling distance and F2 is the second objective or travelling cost.

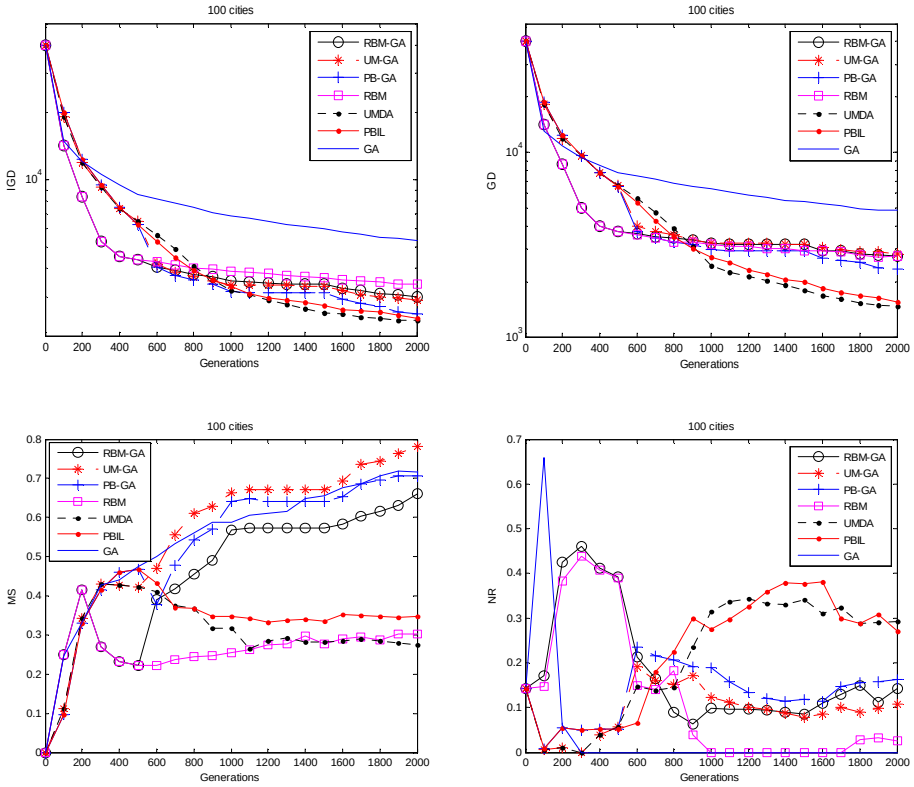


Figure 6.9: Evolution trace of IGD, GD, MS, and NR performance indicators for MOTSP with 100 cities. X-axis is the number of generations and y-axis is the value of the performance indicators.

PB-GA. In order to understand this observation, the results in terms of GD and MS are plotted. The GD results show the parallel observation as IGD. These observations suggest that the EDAs are able to generate a set of closer evolvable solution set to the approximate Pareto optimal front. Since the EDAs and GA implemented in this experimental study utilize similar operators except for the reproduction operators, the good performance of the EDAs is most probably attributed to the incorporation of probabilistic information in guiding the search.

In terms of MS performance indicator, the performance of GA is far better than EDAs. This implies that the diversity preservation in EDAs is poor. This may be attributed to the fact that EDAs only model the certain regions of the promising solutions. Throughout the evolutionary process, the modelled regions may be reduced which results in poor solution diversity preservation. Furthermore, EDAs do not utilize any location information. This may cause the algorithms to be unable to explore the search regions which are further away from the modelled regions. When EDAs is hybridized with a GA, the performance of EDAs in terms of MS is greatly improved, as shown by a higher value of MS obtained by RBM-GA, UM-GA, and PB-GA. This is the main reason why the hybridization of the EDAs and GA is done. EDAs are able to obtain a set of solutions which is closer to the approximate Pareto optimal front; whereas GA proves to be good in terms of maintaining the diversity of the solution set. Thus, the hybridization is expected to complement the strong points of both algorithms so as to overcome their limitations. As for NR performance indicator, it is observed that most of the non-dominated solutions are generated by UMDA and PBIL, and then followed by PB-GA, RBM-GA, UM-GA, and RBM. The performance of GA is the worst.

In order to visualize how the evolvable solutions are distributed in the objective space, Figure 6.8 plots the non-dominated solutions generated by each of the algorithm. For clearer visualization, only 50 randomly picked points of non-dominated solutions are plotted. It is observed that GA is able to produce a set of diverse solutions but they are inferior in terms of proximity. On the other hand, EDAs are able to evolve a set of solutions which is closer to the approximate

Pareto optimal front but the diversity of the solution set is poor. The hybridization of EDAs and GA seems to improve the performance of the algorithms (RBM-GA, UM-GA, and PB-GA) in terms of proximity of the evolvable front to the approximate Pareto optimal front and distribution of the solutions on the approximate Pareto optimal front.

Figure 6.9 shows the evolution trace curves for IGD, GD, MS, and NR performance indicators from the early stages of evolution up to the 200,000 fitness evaluations for the MOTSP with 100 cities. It is observed that RBM has the best convergence speed at the early stages of evolution (in terms of IGD and GD performance indicators). However, the convergence speed of RBM is decreased after about 500 generations. This is most probably caused by the modelling mechanism of RBM in whereby only certain promising regions are being modelled. RBM constructs the probability distribution of the candidate solutions by considering their linkage dependencies. This information is useful in estimating of the search direction, which therefore promotes the convergence speed. Throughout the evolutionary process, the modelled regions will become smaller. Since there is no presence of any enhanced diversity preservation mechanism in RBM, therefore it is hard for the RBM to further explore other promising regions. Eventually, the convergence speed is decreased during the later stage of evolutions. For the other EDAs, the building block of a good schema is harder to build than RBM since only independent probability is considered in their probabilistic modelling. The stochastic recombination of GA also provides fast convergence speed at the early stages of evolution. However, the convergence speed is decreased when it reached to about 100 generations.

In terms of MS performance indicators, the hybrid algorithms exhibit significant change after 500 generations, which is the condition that the EDAs are alternated to GA. The MS values gradually improve throughout the evolutionary process. A higher value of MS indicates that the evolvable solution set is highly distribution along the approximate Pareto optimal front. This is important as poor solution diversity means that most of the optimal tradeoff solutions are not being found. In real-world conditions, a manager will have more options in their decision

making when a set of diverse solutions are provided. In terms of NR performance indicators, it is observed that most of the non-dominated solutions are generated by UMDA and PBIL. This is because both of the algorithms only focus their search within a particular region. Thus, they could find more non-dominated solutions, but poor solution diversity. The hybridization of EDAs and GA improve the ability of the algorithms to further explore and exploit the search space, rectifies the limitation of both of the algorithms.

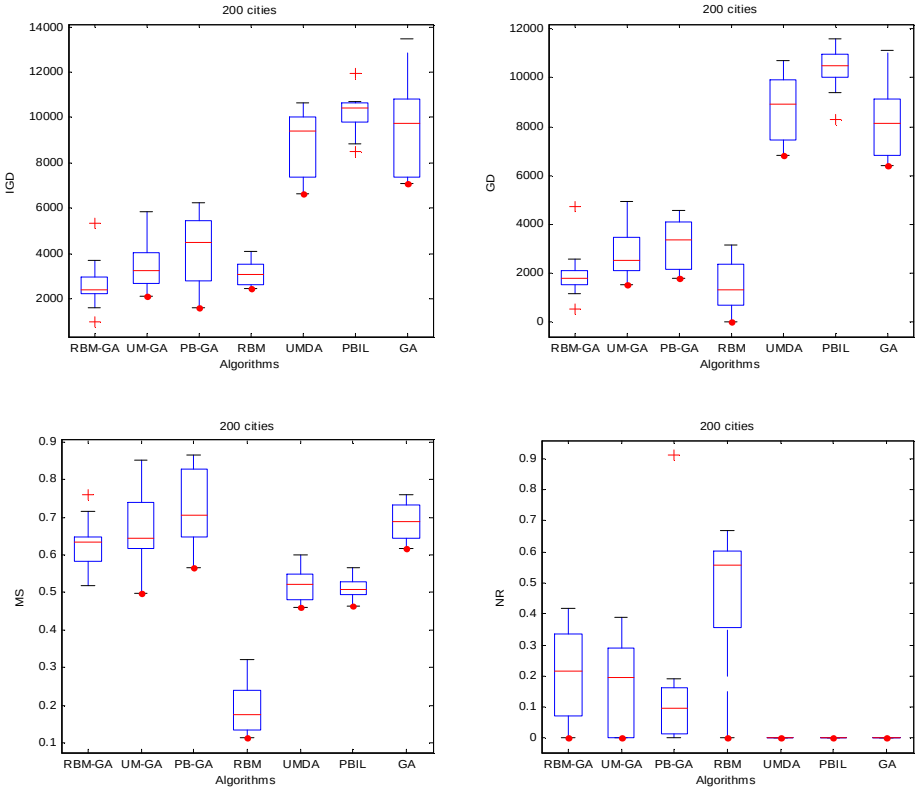


Figure 6.10: Performance metric of IGD, GD, MS and NR after 400,000 fitness evaluations for MOTSP with 200 cities. X-axis is the algorithms and y-axis is the value of the performance indicators.

In order to test the search ability of the algorithms in problem with larger search space, experimental studies were carried out to solve the MOTSP with 200 cities. Performance metric of IGD, GD, MS, and NR at 400,000 fitness evaluations are presented in Figure 6.10. From the performance indicators of IGD and GD, it is observed that the performances of RBM and hybrid algorithms are better as compared to the PBIL, UMDA, and GA. The linkage information is beneficial to the RBM as it helps in the algorithm in the estimation of the probability distribution

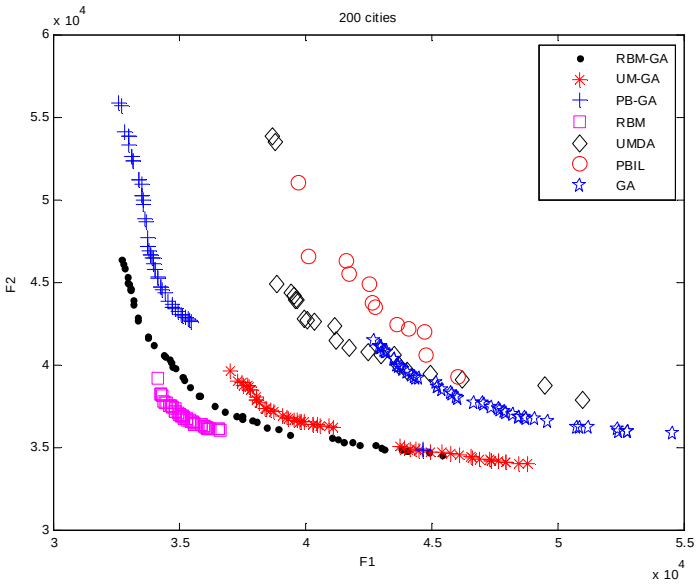


Figure 6.11: Final evolvable front generated by the various algorithms for MOTSP with 200 cities. F1 is the first objective or travelling distance and F2 is the second objective or travelling cost.

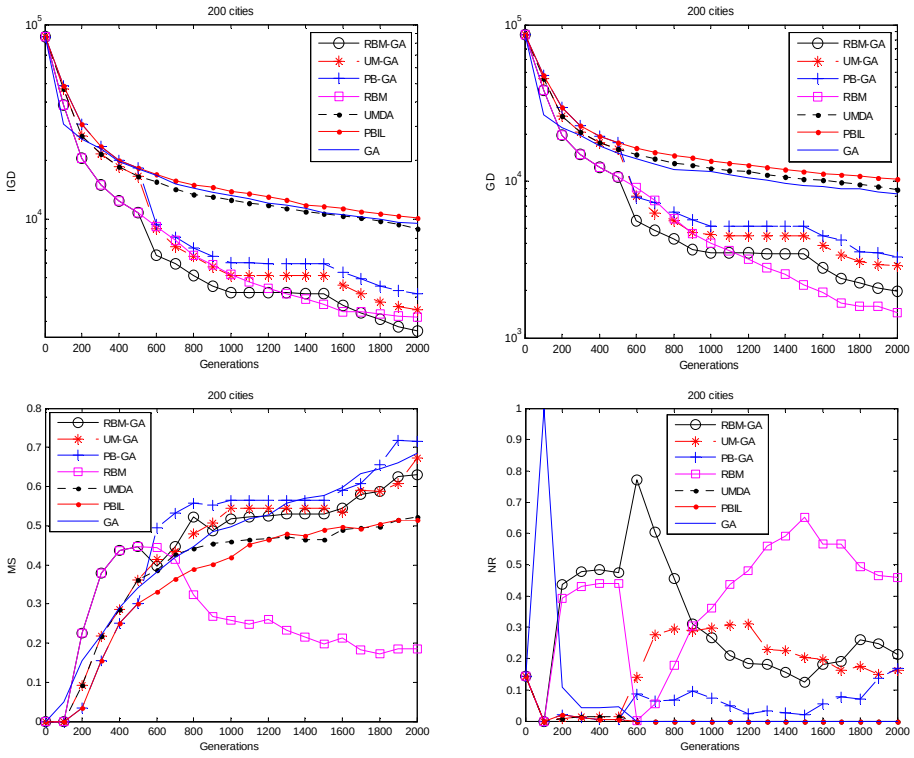


Figure 6.12: Evolution trace of IGD, GD, MS and NR performance indicators for MOTSP with 200 cities. X-axis is the number of generations and y-axis is the value of the performance indicators.

of the cities in the position of the chromosome, and hence this leads to the achievement of better results. This is its main difference with the UMDA and PBIL whereby independent probability

distribution of the cities is being considered. As for GA, the stochastic behaviours of the variation operators are dependent on the chance of recombination, thus it is not possible to predict the direction of the motion. The hybridization of EDAs and GA complement the strong points of each other, and this helps to overcome the limitations of both algorithms, which results in the hybrid being able to outperform the original algorithms. In terms of NR performance indicators, it is observed that RBM evolves most of the non-dominated solutions, and followed by RBM-GA, UM-GA, and PB-GA. However, no non-dominated solution is being generated by UMDA, PBIL, and GA. Overall, RBM performed well in terms of proximity results, but poor solution diversity. This can be further visualized in Figure 6.11, where the final non-dominated evolvable solutions obtained by the various algorithms are plotted.

Figure 6.12 plots the evolution trace of IGD, GD, MS, and NR performance indicators for the MOTSP with 200 cities. Similar observation can be made for RBM, in terms of IGD and MS performance indicators, that it has the fastest convergence speed at the early stages of evolution. Furthermore, the convergence speed of the RBM is enhanced when it is hybridized with GA, as shown in IGD, GD, and MS performance indicators. Generally, in all of the performance indicators, the performance of the hybrid algorithms and RBM are better when compared to the UMDA, PBIL, and GA. In terms of NR performance indicator, GA is only able to maintain a set of non-dominated solutions at the early stages of evolution. However, the NR for GA on the declining trend over the generations. At the end of the evolution, most of the solutions evolved by the RBM dominate the solutions evolved by the other six algorithms.

Performance metric of IGD, GD, MS, and NR for MOTSP with 500 cities are presented in Figure 6.13. It is observed that the performance of RBM-GA, UM-GA, and PB-GA, in terms of IGD, GD, and MS, are better than RBM, UMDA, PBIL, and GA. Among the different hybrid algorithms, RBM-GA and UM-GA have almost similar performance. Other observations that can be noted are that the performance of EDAs is inferior when compared to the GA; and also the performance of PBIL is the worst. In this problem, the search space is huge. When the

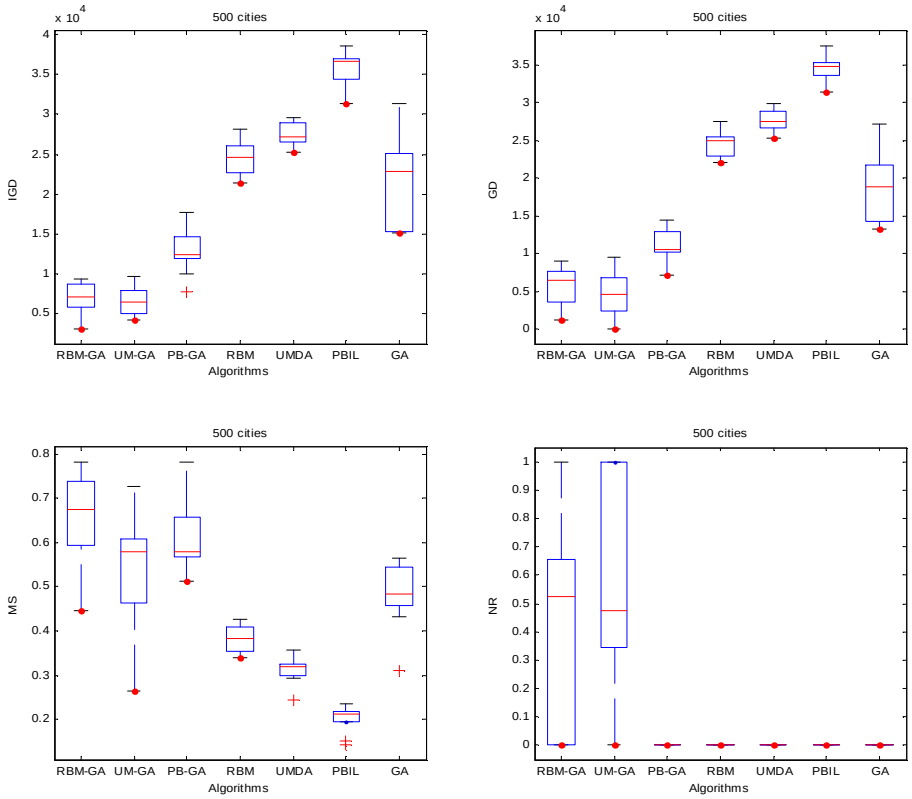


Figure 6.13: Performance metric of IGD, GD, MS and NR after 1,000,000 fitness evaluations for MOTSP with 500 cities. X-axis is the algorithms and y-axis is the value of the performance indicators.

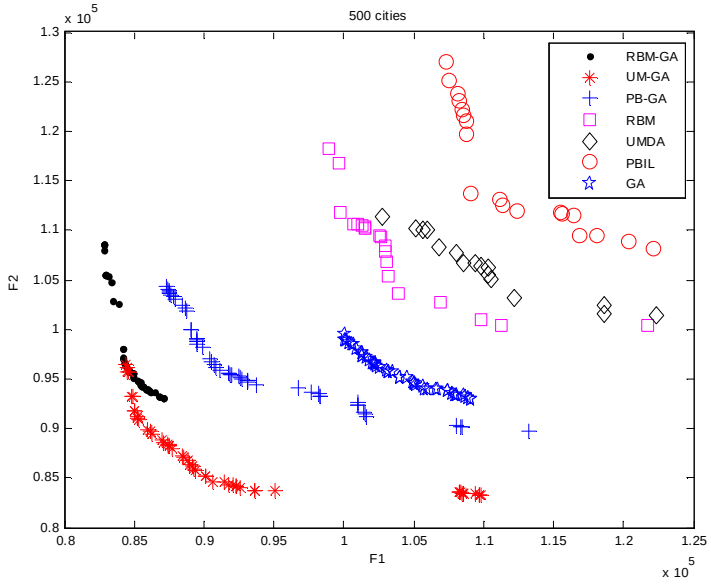


Figure 6.14: Final evolvable front generated by the various algorithms for MOTSP with 500 cities. F1 is the first objective or travelling distance and F2 is the second objective or travelling cost.

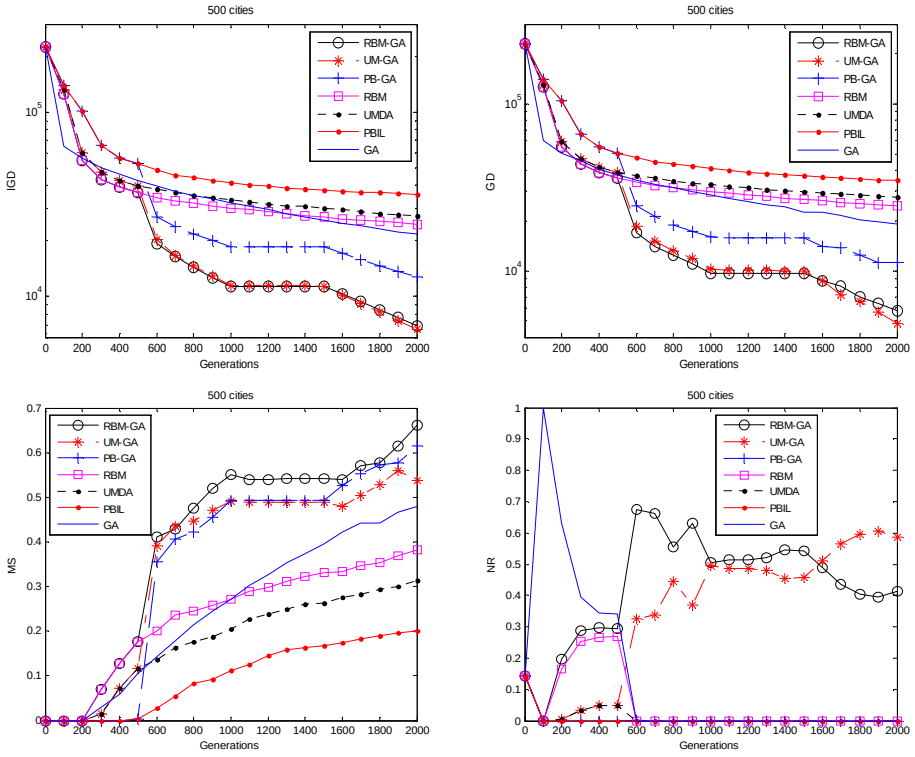


Figure 6.15: Evolution trace of IGD, GD, MS and NR performance indicators for MOTSP with 500 cities. X-axis is the number of generations and y-axis is the value of the performance indicators.

modelling of EDAs only emphasize on certain regions, the diversity of the maintained solution set will be poor. Therefore, this leads to the poor performance of EDAs.

One way to cope with this problem is to divide the promising regions into several clusters and then a probabilistic model is built in every cluster. The clustering can be carried out in decision [88] or objective domain (as presented in Chapter 3). Sampling is subsequently carried out to sample equal number of solutions from each cluster. The generated solutions are then combined to form the new set of offspring. In this chapter, another option to cope with the problem is proposed, which is through the hybridization of EDAs and GA. These results show that the hybridization can enhance the performance of the original algorithms. In terms of MS performance indicator, it is observed that the diversity of the evolvable solution sets on the approximate Pareto optimal front is greatly improved. This is because the EDAs and GA have their own strengths and weaknesses, in which the performance of EDAs is superior in exploring the promising search regions, while GA is superior in searching for a set of diverse solutions. With these features, the

hybridization is able to complement each other so as to overcome their individual limitations, and thus the hybrid algorithms are able to outperform other algorithms. At the end of the evolution, most of the non-dominated solutions are evolved by UM-GA and followed by RBM-GA. As for the other algorithms, they fail to obtain any number of non-dominated solutions. The distributions of the non-dominated solutions evolved by the various algorithms are shown in Figure 6.14.

Figure 6.15 shows the evolution trace of IGD, GD, MS and NR performance indicators for MOTSP with 500 cities. The results obtained in this problem are quite similar to those obtained in problem with 100 and 200 cities. GA has the fastest convergence speed at the early stages of evolution, but the speed is decreased after the algorithm is run up to 100 generations. As for the RBM, its convergence speed outperforms GA when it is run up to 200 generations. Generally, hybrid algorithms are able to achieve better performance, in terms of proximity, diversity, and convergence speed, when compared to the other algorithms. Specifically, RBM-GA and UM-GA display the best performance while the performance of the PBIL is the worst.

6.5.2 Effects of Feasibility Correction Operator on Optimization Performance

The aim of the feasibility correction operator is to refine the sequence in a chromosome in order to guarantee that no repeated visit on any cities in the route of the salesman. Two feasibility correction operators are described in Section 6.3.4, namely permutation correction operator and permutation refinement operator. In permutation correction operator, the unvisited cities are randomly inserted into the positions of the repeated cities. On the other hand, the permutation refinement operator makes use of the available information from the database (travelling distance and travelling cost between the cities) to define the insertion and deletion rules.

In order to investigate the benefit of using the permutation refinement operator instead of the permutation correction operator, the experimental results in terms of IGD performance indicators after running the various algorithms with permutation refinement operator or permutation

correction operator on MOTSP with 100 cities and 200 cities are presented in Table 6.3.

Table 6.3: Performance indicator of IGD after running the various algorithms with permutation refinement operator or permutation correction operator on MOTSP with 100 and 200 cities

Algorithm	100 cities		200 cities	
	P. Refinement	P. Correction	P. Refinement	P. Correction
RBM-GA	**2998.6±964.40**	6878.7±1086.4	**2689.0±1193.9**	19560±1455.9
UM-GA	**2888.1±831.30**	7993.2±1458.4	**3456.0±1101.2**	21448±1648.8
PB-GA	**2504.0±734.50**	7881.7±792.76	**4161.0±1571.9**	20936±4019.7
RBM	**3418.8±718.10**	8391.7±1212.6	**3141.0±587.00**	30030±10673
UMDA	**2349.8±552.70**	8949.1±3862.7	**8955.0±1459.1**	67774±5459.7
PBIL	**2397.5±696.40**	8890.5±3875.4	**10205±982.60**	65984±4161.8
GA	**5371.0±848.26**	7493.6±926.75	**9530.0±2098.5**	15917±2818.2

It is observed that, in all algorithms, the performance of the algorithms with permutation refinement operator outperforms the algorithms with permutation correction operator. This set of results suggests that the utilization of problem domain knowledge is able to enhance the searching task. In the case of EDAs, the diversity of the solution set is exhausted. Therefore, the permutation refinement operator is importance as it serves as a technique to introduce some diverse solutions to the modelled regions.

6.5.3 Effects of Local Search Operator on Optimization Performance

In order to improve the ability of the algorithm to search for better solutions, a local search operator is proposed. This operator will randomly select a sequence of cities, and then relocating them according to the problem domain knowledge.

Figure 6.16 shows the IGD performance indicator obtained by RBM, UMDA, PBIL, and GA for MOTSP with 100 cities under various settings of local search rate (LS=0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0). It is observed that, for RBM, a LS of 0.5 gives the best performance. A lower local search rate (LS=0.1 and 0.3) or a higher setting of LS slightly improve the performance of the algorithm compared to the one without local search (LS=0). For UMDA, the best performance is obtained when LS=0.5. Other settings of LS give random performance. No significant improvement is observed in PBIL. For GA, a larger value of LS gives better result, and the best performance was observed at LS=0.9. Similar to permutation refinement operator, the proposed local search operator also utilizes the problem domain knowledge to guide the search, hence can

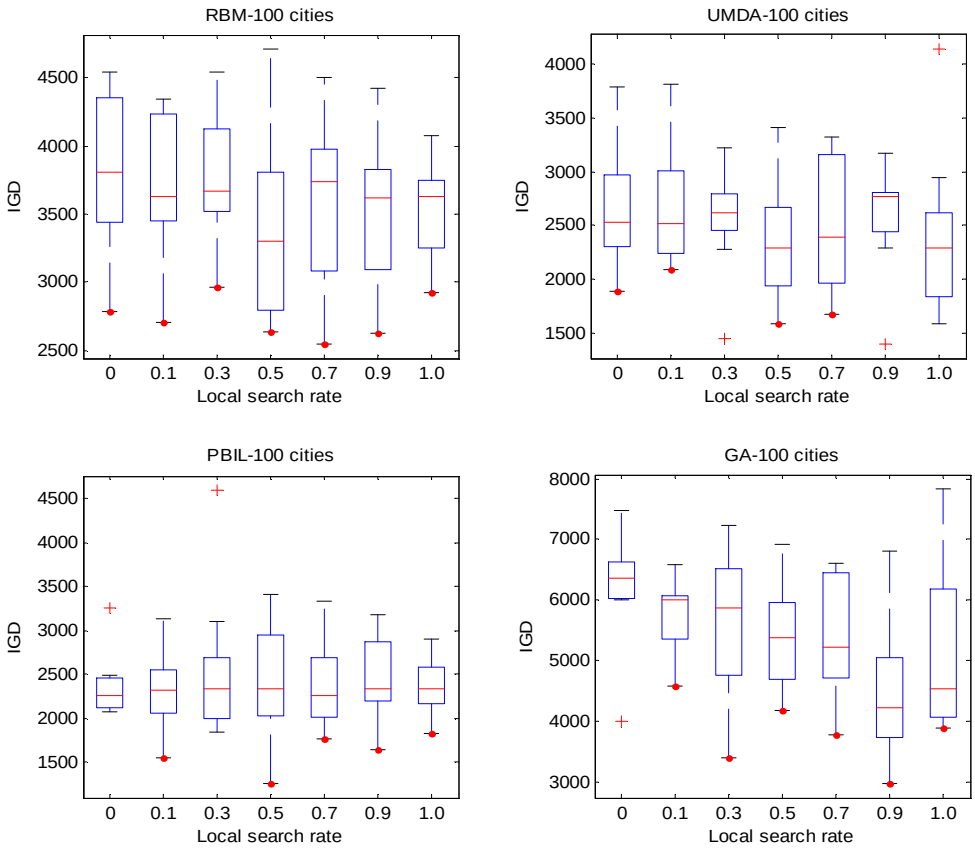


Figure 6.16: Performance indicators of IGD obtained by RBM, UMDA, PBIL, and GA in MOTSP with 100 cities under different settings of local search rate. X-axis is the local search rate and y-axis is the IGD value.

further enhance the search.

6.5.4 Effects of Frequency of Alternation between the EDAs and GA on Optimization Performance

EDAs which model the global distribution of the candidate solutions may be able to drive the search towards certain promising search regions. Since only global information is taken into consideration, the weakness of the algorithms is mainly on its poor exploration of the other search regions which are further away from the modelled regions. Thus, EDAs demonstrate poor solution diversity as indicated in Section 6.5.1. In the literature review, several approaches have been proposed to cope with this limitation. They include the introduction of the mutation operator [89], clustering [88], and Parzen estimator [85] into EDAs.

In the preliminary experimental results, it was observed that GA is able to produce a set of diverse solutions in most of the experimental runs. However, the performance of GA, in terms of proximity of the evolvable solutions to the approximate Pareto optimal front, is inferior when compared to EDAs. This observation sparked that the hybridization of EDAs and GA can create a synergy that can ameliorate the limitation of both algorithms.

In this study, the hybridization is carried out by firstly running the EDAs and then alternates to GA every pre-defined number of generations. Alternation with frequency of every 500 generations is implemented in the above experimental studied. In this section, experimental studies are carried out with various frequency of alternation (fr) including fr =1, 10, 100, 200, 500, 1000, 1500, and 2000. Since the computing budget is set to 2000 generations, which is equal to $2000N$ fitness evaluations, the fr of 2000 means that no alternation from EDAs to GA is occur. Since the aim of the hybridization is to improve the poor solution diversity in EDAs, only results in terms of GD and MS will be looked into and presented.

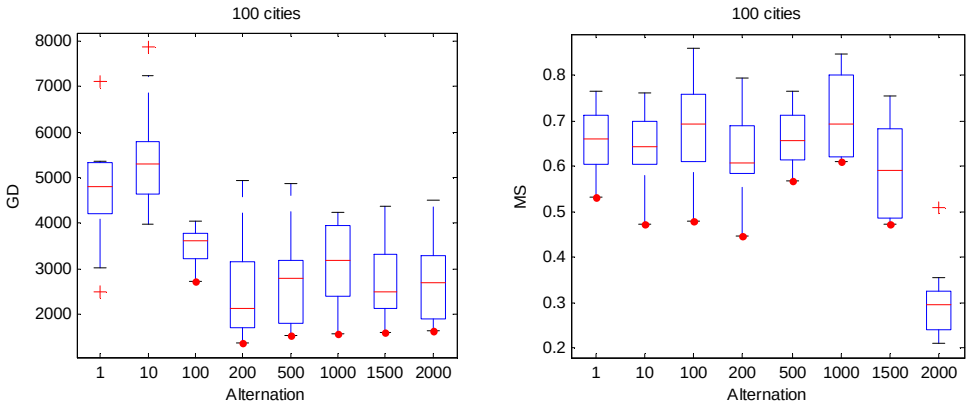


Figure 6.17: Performance indicator of GD and MS obtained by RBM-GA for MOTSP with 100 cities under different settings of the frequency of alternation, fr. X-axis is the frequency of alternation and y-axis is the value of the performance indicators.

Figure 6.17 show the experimental results of GD and MS performance indicators obtained by RBM-GA with respect to the various frequency of alternation fr. In terms of GD performance indicators, it is observed that the performance of the hybrid algorithms are inferior when the fast

alternation are performed (fr=1, 10, and 100). For a larger value of fr ($fr > 100$), the GD performance is better, as indicated by a lower value of GD. This may be attributed to the fact that EDAs and GA may build different building block of schemas during the course of evolution. The fast alternation may prevent the building up of the good genetic schemas; thus, the failure to generate better solutions.

In terms of MS performance indicator, it is observed that the diversity of the solution set obtained by RBM-GA is poor when no alternation is applied (fr=2000). When alternation between RBM and GA are carried out, great improvements are observed in all settings of fr. This observation strengthens the idea that the diversity of the solution set evolved by EDAs can be improved by hybridizing EDAs and GA in the proposed alternating manner. Similar results are also observed in the MOTSP with 200 cities, hence the results will not be presented.

6.5.5 Computational Time Analysis

Comparing between the GA and EDAs, the GA applies genetic operators which only involves cut and insert procedures while EDAs implement the probabilistic modelling approaches to estimate the set of approximate Pareto optimal solutions. Thus, the different optimization schemes adapted in the GA and EDAs will result in different computational time needed. Table 6.4 presents the computational time required by one experimental run of the various algorithms in solving MOTSP with 100, 200, and 500 cities. From the table, the computational time required by the EDAs is higher than the GA. This is due to a higher complexity in construction of the probabilistic models as compared to the genetic operators used in the GA. Among the different EDAs, the RBM surfaces as the most time consuming one. This is attributed to the training stages in RBM which is performed in every generation. As for UMDA and PBIL, they consume lesser computational time than RBM because there is no training in them, and their probabilistic model is directly built from the raw data of the candidate solutions. When the problem size is increased from 100 to 500 cities, all algorithms require additional computational time to complete

one experimental run.

Table 6.4: Computational time (in second) used by the various algorithms for solving MOTSP with 100, 200, and 500 cities

Algorithm	100 cities	200 cities	500 cities
RBM-GA	628.33±14.460	2994.5±7.3088	35486±511.92
UM-GA	27.805±1.0947	194.28±3.1202	2987.4±136.29
PB-GA	27.810±0.8897	199.71±3.7931	3141.1±145.50
RBM	2689.7±91.703	10481±159.82	68792±355.56
UMDA	28.714±0.9036	332.55±8.1735	4648.8±30.824
PBIL	29.167±1.3288	337.84±6.6643	4836.5±25.803
GA	19.005±0.1993	124.98±4.2990	1665.6±43.812

6.6 Summary

This chapter studied the potential of the EDAs and GA in solving the MOTSP with different number of cities. It is among the first attempts to employ EDAs in the study of permutation-based multi-objective optimization problems, specifically the MOTSP. Three EDAs, including UMDA, PBIL, and RBM, have been considered. The EDAs are altered to handle the problem using the integer-number representation. A permutation refinement operator was proposed to make sure the constraint of the problem is fulfilled. In addition, a heuristic-based approach of local search was defined to enhance the search ability of the algorithms. As the limitation of EDAs in evolving a set of solutions with good diversity, EDAs was hybridized with a GA in order to complement their weaknesses. The effectiveness of the seven algorithms was then experimentally studied under the TSP with two objectives and different number of cities.

From the results obtained in the comparative studies, the EDAs was able to achieve better proximity results and GA was superior in maintaining a set of diverse solutions. The hybridization of EDAs and GA mutually complements each other's limitation, thus yielding better optimization performance. Among the different hybrid algorithms, RBM-GA performed the best, and followed by UM-GA, and PB-GA. However, RBM-GA incurred additional computational time in its modelling mechanism. Therefore, if the computational time is not the primary concern, RBM-GA is considered the best algorithms. However, if the computational time is taken

into consideration, UM-GA is preferred.

Since the global Pareto optimal solutions for the problems are unknown, as in the case of most real-world optimization problems, the solution's quality is compared in relation to other solutions obtained from the various algorithms. GA is served as the benchmark algorithm for comparison since the performance of the GA in MOTSP has been proven to be acceptable, if not superb. Therefore, it is concluded that the performance of the EDAs is superior to GA in terms of proximity and convergence speed, and the hybrid algorithms outperform the original algorithms.

Chapter 7

An Advancement Study of REDA in Solving the Multiple Travelling Salesman Problem

The multi-objective multiple travelling salesman problem (MmTSP) is a generalization of the classical multi-objective travelling salesman problem. In this chapter, a formulation of the MmTSP, which considers the weighted sum of the total travelling costs of all salesmen and the highest travelling cost of any single salesman, is proposed. REDA in the decomposition-based framework of multi-objective optimization is developed and used to solve the formulated problem. Due to the limitation of REDA in generating a wide range of solutions, the REDA is hybridized with the evolutionary gradient search. Simulation studies are carried out to examine the optimization performances of the proposed algorithm on MmTSP with different number of objective functions, salesmen, and problem sizes. The effectiveness and efficiency of the algorithms are tested and benchmarked against several state-of-the-art multi-objective evolutionary paradigms.

7.1 Introduction

The multiple travelling salesman problem (mTSP), where multiple salesmen are involved in the routing in order to achieve a common goal, is a generalization of the classical TSP. In the mTSP, Ω salesmen are instructed to visit n cities ($\Omega < n$), whereby all the salesmen will start from and end at the single depot (may be multiple depots) after visiting the ordered cities. Each city can only be visited once, and the total cost for all salesmen is required to be minimized. The cost can be defined as distance, time, expense, risk, etc. When $\Omega = 1$, the problem simplifies to the classical TSP. The mTSP is more complex than the TSP since it is required to allot a set of cities to each salesman in an optimal ordering while minimizing the total cost for all salesmen. However, the mTSP is more appropriate for real life problems where more than one salesman is usually involved. The problem is closely related to the school bus routing problem [203], design of global navigation satellite system [204], interview scheduling [205], hot rolling scheduling [206], mission planning [207], etc. Over the past few decades, research on the TSP has attracted a great deal of attention. However, the mTSP has not received the same amount of research effort compared to the TSP. Due to the high complexity of the mTSP (NP hard problem [208]), exact algorithms are unsuitable to solve the problem even for a moderate number of cities. Even though heuristic approaches are unable to guarantee optimal solutions, they are still able to obtain approximate optimal solutions within specific time or computational constraints.

In addition, many real life scheduling problems also involve several conflicting objectives that have to be simultaneously optimized. In the evolutionary multi-objective framework [1], no single point is an optimal solution. Instead, the optimal solution is a set of non-dominated solutions, which represents the tradeoff between the multiple objectives. In this case, fitness assignment to each solution in the evolutionary framework is considered as an important feature for the assurance of the survival of fitter and less crowded solutions to the next generation. Much research has been carried out over the past few decades to address this issue, and fitness assignment based on domination is one of the most popular approaches [32]. However, this fitness

assignment approach is less efficient in solving many-objective problems. This is because the strength of the domination is weakened when there are many objectives, which in turn results in poor decision making in the selection of promising solutions. Recently, the classical approach for multi-objective optimization based on aggregation (the domination-based approach) is re-formularized into a population-based approach [71, 74], whereby a set of non-dominated solutions is obtained from a single simulation run. In the decomposition-based approach, it is not required to differentiate the domination behaviour of the solutions. Instead, the solutions are aggregated according to any classical aggregation approaches.

This chapter studies the hybridization of an evolutionary optimizer with a local search technique in the decomposition-based framework of multi-objective optimization to deal with the multi-objective mTSP (MmTSP). Firstly, REDA in the decomposition-based framework of multi-objective optimization is developed. The performance of the algorithm is then enhanced by hybridizing REDA with the evolutionary gradient search (EGS). A new formulation of the objective functions for the mTSP is proposed and extended to the multi-objective framework. The proposed algorithms are then used to solve the formulated problem and simulation studies are carried out on various instances of the problems with different number of objective functions, salesmen, and cities. The proposed algorithms are then rigorously compared with several state-of-the-art evolutionary multi-objective optimizers.

The remaining parts of the chapter are organized as follows. The following section presents a literature review on the application of EAs in mTSP, as well as a brief introduction of the EGS. Section 7.3 describes the problem formulation of the MmTSP, while the proposed algorithm is highlighted in Section 7.4. Section 7.5 presents the implementation that will be carried out. Section 7.6 presents the experimental results and discussions. Conclusion is drawn in the last section.

7.2 Background

In this section, a literature review, focusing on the application of evolutionary approaches to mTSP, is provided. Readers are referred to [209] for a more rigorous review of the works that involved non-evolutionary techniques, which is out of the scope of this chapter. A brief introduction on the EGS will also be presented.

7.2.1 Existing Studies

The first implementation of genetic algorithms (GAs) to solve the mTSP was carried out in [210]. Zhang *et al.* used a simple GA with natural representation to schedule the multiple teams of photographers to visit a large number of elementary and secondary schools. The objective is to minimize both the total distance traveled and the time consumed, such that the time constraints are satisfied and each team must be able to visit at least two schools daily. Unfortunately, the authors did not elaborate on how they manipulated multiple teams in their chromosome representation.

Another application of the mTSP, involving a hot rolling scheduling problem in Shanghai Baoshan Iron and Steel complex, was studied by Tang *et al.* [206]. The problem considers actual production constraints and aims to schedule multiple turns within the same shift. The authors first modelled the hot rolling scheduling problem as an mTSP. Next, the mTSP was converted into a classical TSP through the proposal of a one-chromosome representation. The selection operation was modified such that the best solutions obtained so far were always selected to be one of the parent chromosomes to undergo crossover operation.

In [211], the author studied the vehicle scheduling problem using GAs. Since there are multiple vehicles involve in the routing, the problem can basically be classified as an mTSP. In the chapter, a two-chromosome representation was proposed. The first chromosome locates the cities while the second chromosome indicates which vehicle is to be assigned to visit the city specified in the first chromosome. In [212], a two-part chromosome representation was proposed for the mTSP. Under this scheme, the chromosome for a gene is divided into two distinct parts.

The first part of the chromosome allots the permutation of the cities, while the second part of the chromosome determines the number of cities to be visited by each salesman. This means that there will be an additional of Ω genes in the chromosome for Ω salesmen. In this two-part chromosome representation, the solution space is also smaller than the two-chromosome representation. The results also demonstrated that the proposed representation is able to produce better results than the one-chromosome representation under most of the test instances.

Zhao *et al.* [213] proposed a GA, which utilized the one-chromosome representation, to solve the mTSP. The algorithm employed a pheromone-based crossover operator which utilized information of edge lengths, adjacency relations, and pheromone levels to construct new solutions. Several local search strategies (relocation, exchange, and 2-opt) were also used to facilitate the search. The grouping GA was also used by Singh and Baghel [208] who proposed a replacement policy to reduce problem redundancy. In their work, two different objective functions were considered - minimizing total distance traveled by all salesmen and minimizing the maximum distance traveled by any salesman.

Recently, multi-chromosome representation of mTSP solutions was proposed in [214] where the route assigned to each salesman was represented in a chromosome. Therefore, each solution has Ω associated chromosomes. The crossover and mutation operators designed to deal with the representation were also proposed.

7.2.2 Evolutionary Gradient Search (EGS)

Local search, which has been proven to be able to improve the performance of global search, especially EAs [215, 216], is used to exploit the local optimal in a specific region. In this chapter, the EGS are studied and their brief descriptions are as follows (consider the minimization case).

Gradient search is one of the classical continuous optimization approaches. The direction, in terms of gradient, is captured and is used to guide the search. Every simulation run will generate a single solution. Recently, this approach has been adapted into evolutionary mecha-

nism through the introduction of the population-based and survival-of-the-fittest concepts into the algorithm so as to produce evolutionary gradient search (EGS) [138, 217, 218]. In this approach, multi-directional searches are carried out. The gradient is the direction calculated from the evolutionary movement instead of the single movement of a solution. The pseudo-code of the multi-point EGS can be found in Figure 7.1. Firstly, initial step size σ_0 is predetermined. Step size σ is used to govern the degree of mutation applied to generate local neighbours and offspring. After selecting an individual as the initial solution, L local neighbours are generated by perturbing the initial solution using mutation with normal distribution of zero mean and σ^2 variance. The global gradient direction is subsequently estimated from the local neighbours as presented in Step 5 of Figure 7.1. It is to be noted that the gradient offspring is generated during Step 6, and a factor ε is used to control the mutation step size as shown in Step 7. The solution is updated in Step 8 and the process is repeated until the stopping criterion is met.

Begin

1. Input: Define initial step size σ_0 and set $t = 1$

Do while ("Stopping criterion is not met")

For $j = 1$: K(Number of solutions undergoing local search)

2. Initial solution: Select a solution $\boldsymbol{x}^j$ from selection pool
3. Reproduction: Create L local neighbors $\boldsymbol{y}^i$, $i \in (1,2,\dots,L)$ by perturbing $\boldsymbol{x}^j$ using normal mutation $N(0, \sigma_t^2)$
4. Evaluation: Calculate the fitness value of $\boldsymbol{y}^i$, $\boldsymbol{f}(\boldsymbol{y}^i)$
5. Direction: Estimate the global gradient direction as follows

$$\hat{\boldsymbol{v}} = \frac{\sum_{i=1}^{L}[\boldsymbol{f}(\boldsymbol{y}^i) - \boldsymbol{f}(\boldsymbol{x}^j)](\boldsymbol{y}^i - \boldsymbol{x}^j)}{\left\|\sum_{i=1}^{L}[\boldsymbol{f}(\boldsymbol{y}^i) - \boldsymbol{f}(\boldsymbol{x}^j)](\boldsymbol{y}^i - \boldsymbol{x}^j)\right\|}$$

6. Create offspring:

$$\boldsymbol{g} = \boldsymbol{x}^j - \sigma_t \hat{\boldsymbol{v}}$$

7. Update mutation step size σ_{t+1}:

$$\sigma_{t+1} = \begin{cases} \sigma_t \varepsilon & \text{if } \boldsymbol{f}(\boldsymbol{g}) < \boldsymbol{f}(\boldsymbol{x}^j) \\ \sigma_t / \varepsilon & \text{otherwise} \end{cases}, \varepsilon = 1.8$$

8. Update solution: if $\boldsymbol{f}(\boldsymbol{g}) < \boldsymbol{f}(\boldsymbol{x}^j)$ then $\boldsymbol{x}^j = \boldsymbol{g}$
9. Output: Output $\boldsymbol{x}^j$

End for j

$t = t + 1$

End do

End

Figure 7.1: Pseudo-code of the evolutionary gradient search algorithm

7.3 Proposed Problem Formulation

The mTSP is the generalization of the classical TSP (single salesman), where Ω salesmen are involved in the routing. The aim is to minimize the total travelling cost of all the salesmen under the constraints that each city must be visited strictly once by any salesman, and the salesman must return to the starting depot after visiting his final city. The travelling cost could be defined as the travelling distance, travelling time, travelling expense, travelling risk, etc incurred. In MmTSP, two or more of the travelling cost (treated as objective functions) will be optimized simultaneously. Each salesman will have his own route and there should be no repeated visit on any cities in the route of the salesman.

In the literature, the aim of the mTSP is specified to be either minimizing the total travelling cost of all salesmen or the highest travelling cost incurred by any single salesman [219]. It is to be noted that the aim of the mTSP is different from the objective functions concerned. For example, the aim of the mTSP can be set to minimizing the total travelling cost of all salesmen while the objective functions (travelling cost) considered are travelling distance and travelling risk. On the other hand, the aim of the mTSP can be set to minimizing the highest travelling cost incurred by any single salesman while the objective functions (travelling cost) considered are similar to the previous example. In this chapter, the focus is tailored specifically for the mTSP with single depot; considering the minimization of the total travelling cost and the balancing of the workload among all salesmen. This is achieved by formularizing the objective function to be the weighted sum of the total travelling cost of all salesmen and the highest travelling cost by any single salesman. In the context of multi-objective optimization, more than one objective is subject to be minimized, which can be formulated as follows:

Minimize:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), ..., f_m(\mathbf{x}))$$

$$f_1(\mathbf{x}) = \omega_1 \times TC^1 + \omega_2 \times MC^1$$

$$\vdots$$

$$f_m(\mathbf{x}) = \omega_1 \times TC^m + \omega_2 \times MC^m$$

where

$$TC^k(\mathbf{x}) = \sum_{j=1}^{\Omega} IC_j^k(\mathbf{x})$$

$$MC^k(\mathbf{x}) = \max_{1 \leq j \leq \Omega} (IC_j^k(\mathbf{x}))$$

$$IC_j^k(\mathbf{x}) = \left(\sum_{i=1}^{n_j - 1} D^k(a_{i,j}, a_{i+1,j}) \right) + D^k(a_{n_j,j}, a_{1,j})$$

In the above formulation, $\mathbf{x} = (x_1, x_2, ..., x_n)$ is the decision vector, $\mathbf{x} \in \mathbb{R}^n$, $\mathbb{R}^n$ is the decision space, $a_{i,j}$ is the i^{th} visiting city by salesman j, m is the number of objective functions, ω_1 and ω_2 are the weights to balance between total cost and highest cost ($\omega_1 + \omega_2 = 1.0$), TC is the total travelling cost of all salesmen, MC is the highest travelling cost of any single salesman, IC is the individual travelling cost, Ω is the number of salesmen, n_j is the number of cities traveled by salesman j, $D^k(a_{i,j}, a_{i+1,j})$ is the travelling cost (for the k^{th} objective function) between cities at locations i and $i + 1$ for salesman j. In a chromosome, two conditions should be met, which are all the cities must be visited exactly once and each salesman must be assigned at least one city in his travelling route.

7.4 A Hybrid REDA with Decomposition

The proposed algorithm, named as hybrid REDA (hREDA), consists of four main mechanisms. They are chromosome representation, decomposition, modelling, and local search. The hREDA is developed in the decomposition-based framework of multi-objective suggested in [74].

7.4.1 Solution Representation

In the implementation stage, one-chromosome representation [206] is utilized to represent the order of the cities to be traveled by Ω salesmen. This scheme introduces $\Omega - 1$ pseudo cities (integer values <0) to the chromosome. These pseudo cities represent the same initial city where all the salesmen will start their routes. Therefore, each chromosome may consist of $n + \Omega - 1$ genes. An example of the chromosome representation with nine cities and three salesmen is illustrated in Figure 7.2. The sequence of travel is as follows. The first salesman starts from the initial city 0 then visits cities 2, 5, and 7 in that order. The second salesman again starts from the initial city (city indicated by -1) then visits cities 1 and 8 in that order. The last salesman starts from the initial city (city indicated by -2) then visits cities 6, 4, and 3 in that order.

Figure 7.2: One-chromosome representation

7.4.2 Algorithmic Framework

In the decomposition-based framework, the fitness assigned to each solution can be based on any classical aggregation approaches. In this chapter, the Tchebycheff approach is used and hREDA is described according to this approach. A set of evenly distributed weight vectors $\boldsymbol{\lambda}^1, ..., \boldsymbol{\lambda}^T$ and the reference point $\mathbf{z}^*$ are generated, where T is the number of subproblems. The algorithm decomposes the population into T scalar optimization subproblems according to the Tchebycheff formulation and the fitness value of the j^{th} subproblem is defined as the following equation:

$$gt(\mathbf{x}|\boldsymbol{\lambda}^j, \mathbf{z}^*) = \max_{1 \leq x \leq m} \left\{ \lambda_i^j | f_i(\mathbf{x}) - z_i^* | \right\} \tag{7.1}$$

The pseudo-code of hREDA is presented in Figure 7.3. In Step 1, the Q neighbours (denoted

Inputs:
- Computing budget or stopping criterion
- Population size or number of subproblems, T
- A set of uniformly distributed weight vectors used in decomposition, $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^T$
- Number of neighbors of each weight vector, Q
- Number of traveling salesmen, Ω
- Number of local neighbors in local search, L
- Number of cities, n

Output:
- A set of solutions generated by the optimizer in both decision and objective space

Step 1: Initialization:
a) Compute the Euclidean distance between all the weight vectors and then group the Q closest weight vectors $B(i) = \{i_1, .., i_Q\}$, $i \in [1, T]$, to each weight vector. T is the number of subproblems which is identical to the population size N
b) Randomly generate the initial population $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N$ in integer number from $[1 - \Omega, n - 1]$. No integer number is repeated. Set $FV^i = \boldsymbol{f}(\boldsymbol{x}^i)$
c) Initialize $\boldsymbol{z}^* = (z_1^*, \ldots, z_P^*)$ by setting $\boldsymbol{z}^*$ according to the minimum objective value of the initial population

Step 2: Reproduction based on REDA:
a) Decode the integer-number representation of the cities into the binary-number representation. Train the network. Encode the binary-number representation of cities into integer-number representation. Construct the probabilistic model $p_g(\boldsymbol{x})$ by computing the marginal probability of each city $(c_1, .., c_P)$, where $P = n + \Omega - 1$, in each permutation location
b) Sample $p_g(\boldsymbol{x})$ to generate N children solutions
c) Improvement: Apply specific heuristic approach to repair the chromosomes to ensure that the conditions of routing are satisfied. Penalize the solution if any salesman is not assigned any city by multiplying the objective value with a constant ϑ, $\boldsymbol{f}(\boldsymbol{y}) = \vartheta \times \boldsymbol{f}(\boldsymbol{y})$

Step 3: Update solution:
For $i = 1, \ldots, T$, do
a) Update of $\boldsymbol{z}^*$: For $j = 1, \ldots, m$, if $z_j^* > f_j(\boldsymbol{y}^i)$, then set $z_j^* = f_j(\boldsymbol{y}^i)$
b) Update of neighboring solutions: For $j \in B(i)$, if $gt(\boldsymbol{y}^i | \boldsymbol{\lambda}^j, \boldsymbol{z}^*) \leq gt(\boldsymbol{x}^j | \boldsymbol{\lambda}^j, \boldsymbol{z}^*)$, then set $\boldsymbol{x}^j = \boldsymbol{y}^i$ and $FV^j = \boldsymbol{f}(\boldsymbol{y}^i)$
End do

Step 4: Local search:
a) Perform local search if local search is activated. Next, apply Step 3 to update the created children solutions.

Step 5: Stopping criterion: If stopping criterion is met, then stop. Else, go to Step 2

Figure 7.3: Pseudo-code of hREDA

as $B(i) = \{i_1, ..., i_Q\}$) that are nearest, in terms of Euclidean distance, to each weight vector are determined. Then, initial chromosomes in the form as indicated in Figure 7.2 are randomly generated. Objective values $\mathbf{f}^1(\mathbf{x}), ..., \mathbf{f}^N(\mathbf{x})$, N is the population size, are calculated based on the formulation in Section 7.3, and the reference point $\mathbf{z}^*$ is set to the minimum objective value of the initial population. It is to be noted that $N = T$ is implemented in this chapter as suggested in [74, 76]. This mean that each subproblem is represented by a single solution in the population. In Step 2, the integer-number representation of the cities are decoded into the binary-number representation. An RBM is trained using the CD training mechanism [119] in order to obtain the trained weights, biases, and hidden states of the network. After training, an encoding

scheme is performed to obtain the probability of the cities in each position of the chromosome. The procedures of decoding, training, and encoding are identical to the procedures discussed in Section 6.3.3. The probability of the cities is then computed by calculating the probability of existence of each city in each permutation location in the chromosome. In the model, a $P \times P$ probabilistic matrix $p_g(x_i)$, where $P = n + \Omega - 1$, is constructed as follows:

$$p_g(\mathbf{x}) = \begin{bmatrix} p_g(x_1 = c_1) & \dots & p_g(x_P = c_1) \\ \vdots & \ddots & \vdots \\ p_g(x_1 = c_P) & \dots & p_g(x_P = c_P) \end{bmatrix} \tag{7.2}$$

$$p_g(x_i = c_j) = \frac{\sum_{k=1}^{N} \delta_k(x_i = c_j) + \frac{Z_i}{P \times N}}{Z_i + \frac{{Z_i}^2}{P \times N}} \tag{7.3}$$

where $p_g(x_i)$ is the probability distribution of the cities at generation g, $p_g(x_i = c_j)$ is the probability of city j to be located at the i^{th} position of the chromosome, c_j is the city j ($c_1 = 1 - \Omega, ..., c_P = n - 1$) and Z_i is the normalizing constant as indicated in equation 6.8.

N new solutions are subsequently generated by sampling the constructed probabilistic model as follows:

$$y_j = \begin{cases} c_1 & \text{if random } (0,1) \leq p_g\left(x_j = c_1\right) \\ c_2 & \text{if } p_g(x_j = c_1) < \text{random } (0,1) \leq \sum_{i=1}^{2} p_g(x_j = c_i)) \\ & \vdots \\ c_P & \text{if } \sum_{i=1}^{P-1} p_g(x_j = c_i) < \text{random}(0,1) \leq \sum_{i=1}^{P} p_g(x_j = c_i) \end{cases} \tag{7.4}$$

where y_j is a newly generated city at j^{th} position of a chromosome, and random (0,1) is a randomly generated value between [0, 1].

Since the sampling is carried out marginally, the existing cities in a chromosome are not taken into consideration by the sampling mechanism. Therefore, some cities may appear more

than once while some cities may not even be included in a chromosome. The chromosome is repaired according to the heuristic approach proposed in the previous chapter to ensure that there is no repetition of any cities. In this approach, those repeated and unallocated cities are determined. The unallocated city is inserted to the location of the repeated city if the unallocated city has the smallest travelling cost (e.g. distance, times, charge, risk, etc) to the adjacent cities in the location of the repeated city. For example, let assumes that a salesman is instructed to visit 5 cities in an order of 1, 3, 2, 3, 5. In this case, city 3 is repeated while city 4 is not included in the order. The travelling costs of visiting cities in an order of 1, 4, 2 and 2, 4, 5 are calculated. If it is in the condition that the travelling cost of visiting cities in the order of 1, 4, 2 is smaller than the cost of visiting cities in the order of 2, 4, 5, then the former order is applied. Thus, the final sequence of travel is 1, 4, 2, 3, 5.

In the event that some salesmen are not being assigned any cities in their routes, there will be a penalty added to the objective values of the solutions by multiplying the original values with a constant value ϑ. This is done to weaken the solutions since the workloads assigned to some of the salesmen in those solutions are unevenly distributed. In the implementation stage, $\vartheta = 10$ is applied. Step 3 updates $\mathbf{z}^*$ followed by FV. For $\mathbf{z}^*$, it is the reference point used in the Tchebycheff approach and is updated by taking the minimum value of the objective functions. It is emphasized that Step 3b is one of the important features in the decomposition-based framework whereby the fitness of the solutions are assigned according to equation (7.1). All solutions sampled from the probabilistic model will also be updated one by one to all neighbouring solutions, and the superior solutions will replace the inferior ones.

Local search is performed if it is activated and it is applied every generation thereafter. The same procedure is carried out until the stopping criterion is met. The pseudo-code of the EGS is presented in Figure 7.1. However, some modifications are required to adapt the EGS for the permutation-based problem. Firstly, each subproblem that undergoes local search will be perturbed to generate L local neighbours, which are created by simply swapping two genes in a

chromosome. For each local solution, Step 3 is carried out to update the $\mathbf{z}^*$ and FV. The fitness of the solutions are aggregated according to the weighted sum approach by using the available $\boldsymbol{\lambda}^1, ..., \boldsymbol{\lambda}^S$ values. This is different from the previous implementation [138, 217, 218] where the weights are randomly generated. The global gradient direction is estimated according to Step 5 of Figure 7.1. In Step 6 of Figure 7.1, we have $\mathbf{x}^j$ as a vector of floating values. However, in MmTSP, $\mathbf{x}^j$ is the vector of cities which may not be suitable for creating an offspring. As such, the average cost between the local neighbours to the original chromosome is calculated and then assigned to $\mathbf{x}^j$. Following which, $\mathbf{g}$ is updated. However, $\mathbf{g}$ is a vector of floating values and it cannot be used to represent a vector of cities. Thus, we have to determine the candidate city to be the one that has the closest $\mathbf{g}$ cost value to the original city. For example, let $g_1 = 100$, city 1 as the original city, and cities 2 and 3 as other cities that will be visited. If the travelling cost between cities 1 and 2 is 200 and the travelling cost between cities 1 and 3 is 150, then city 3 is the candidate city because the travelling cost is closer to $\mathbf{g}$ value or specifically g_1 value. The mutation step size is updated according to Step 7 of Figure 7.1, and the gradient solution is updated to the population according to Step 3 of Figure 7.3. A shared mutation step size σ_t is used to generate the gradient offspring and is adaptively tuned based on the quality of the estimated gradient. $\sigma = 1.8$ is used as suggested in the previous work [138, 217, 218].

The diversity of the solutions is maintained through the preselected weight vectors. The idea is similar to that in classical aggregation algorithms whereby multiple weight settings are used to produce an estimated optimal solution for each weight setting. However, the decomposition MOEA maintains a set of solutions in each simulation run rather than carrying out multiple runs as seen in classical aggregation approaches. Even though there is no obvious elitism being applied, it is implicitly presented in Step 3b of Figure 7.3, where Q nearest neighbours (parents) are updated by comparing their fitness values with those of the offspring.

7.5 Implementation

Table 7.1: Parameter settings for experiments

Parameter	Setting
Population size, N	Number of cities, n
Number of subproblems, T	N
Number of cities, n	100, 300, 500
Number of salesmen, Ω	2, 5, 10 for 100 cities; 2, 5, 10, 30 for 300 cities; 2, 5, 10, 20, 50 for 500 cities
Number of objective functions, m	2 and 5
Number of local neighbours, L	10
Computing budget in fitness evaluations	200,000 for 100 cities; 600,000 for 300 cities; 1,000,000 for 500 cities
Crossover and mutation rate in NSGA-II and MOEA/D	0.8 and 0.005
Initial step size (σ_0) and ε in EGS	300 and 1.8
Local search activation (in terms of fitness evaluations)	After 100,000 for 100 cities; 300,000 for 300 cities; 500,000 for 500 cities
Number of hidden units and training epochs	10 and 2

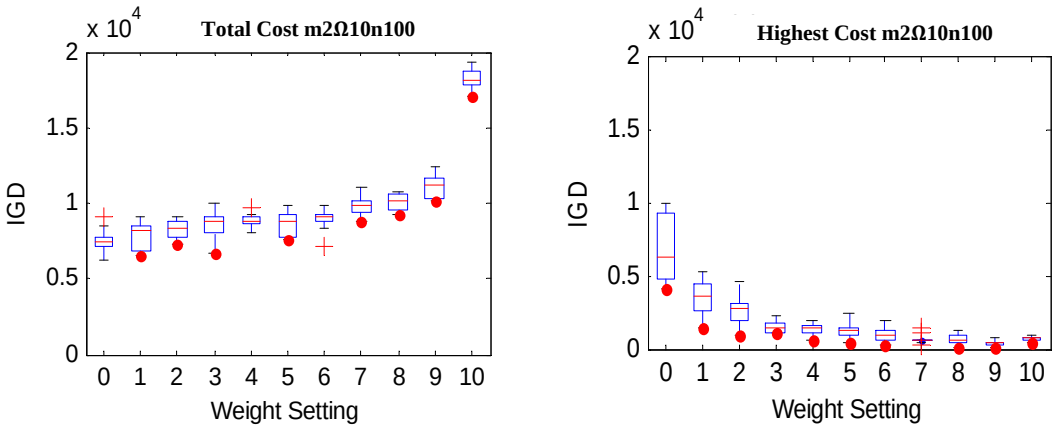
All the algorithms in this study were implemented in C++. The experimental settings are shown in Table 7.1. MmTSP with two and five objectives are studied. A $n \times n$ cost matrix is randomly generated for each objective in the range of [0, 1000] [9,192]. For experimental studies, the results are compared based on the performance metrics of inverted generational distance (IGD) [201] and Pareto front. A smaller IGD value implies better proximity and spread. Since the optimal solution set to the problem is unknown, the estimated optimal front is formed using the non-dominated solutions found from all the algorithms and all simulation runs.

Five algorithms are put to comparison. hREDA is the proposed hybrid algorithm. REDA/D is a pure decomposition-based REDA which is directly modified from the previously developed domination-based REDA. MOEA/D is a pure decomposition-based GA proposed in [74] and used in [192]. UMGA is a synthesizing algorithm between EDA and NSGA-II proposed in the previous chapter. Lastly, NSGA-II is one of the most popular MOEAs based on the concept of domination proposed in [32] and used in the previous chapter. The results presented have been averaged over 10 independent runs with different random number seeds. The test instances consist of 24 MmTSP with different number of objective functions (m), salesmen (Ω), and problem size (n). The problem is denoted in the form of m2Ω5n100, which refers to an MmTSP with two objective functions, five salesmen, and 100 cities.

7.6 Results and Discussions

7.6.1 Effects of Weight Setting on Optimization Performance

The formulation of the MmTSP in this chapter takes into account the weighted sum of total travelling cost of all salesmen and the highest travelling cost of any single salesman. The weight setting is dependent on the preference of the manager whether he wants to achieve the lowest total traveling cost of all salesmen or he wants to achieve the balancing of workload of all salesmen. If the aim is to obtain the lowest total travelling cost, the weights will be set to $\omega_1 = 1.0, \omega_2 = 0.0$. On the other hand, if the final aim is to balance the workload of all salesmen, the weights will then be set to $\omega_1 = 0.0, \omega_2 = 1.0$. However, if the aim is to achieve tradeoff between the two aims, then different weight settings should be employed.



(a) (b)

Figure 7.4: IGD metric for (a) total travelling cost of all salesmen and (b) highest travelling cost of any single salesman under various weight settings for the MmTSP with two objective functions, 10 salesmen, and 100 cities (m2Ω10n100)

Table 7.2: Indices of different weight settings

Index	0	1	2	3	4	5	6	7	8	9	10
ω_1	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0
ω_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Simulations were carried out to study the performance of hREDA under various weight

settings. Figure 7.4 shows the IGD metric for (a) total travelling cost of all salesmen and (b) highest travelling cost of any single salesman under various weight settings for the MmTSP with two objective functions, 10 salesmen and 100 cities (m2Ω10n100). The indices of the weight settings are illustrated in Table 7.2. From Figure 7.4(a), it is observed that the algorithm is able to produce solutions with minimum total travelling cost of all salesmen under the weight setting of $\omega_1 = 1.0, \omega_2 = 0.0$. This is expected since the weight setting deems solutions with smaller total travelling cost as superior. However, Figure 7.4(b) shows that the weight setting causes imbalance workload for the salesmen. On the other hand, if the focus is to minimize the highest travelling cost of any single salesmen (index 10), it is then observed that the highest cost is far smaller than that of index 0, but this leads to the solutions having the highest total travelling cost. This observation suggests that there is a tradeoff between both aims. Striking a balance between both aims can be achieved by setting the weights to intermediate values, which will lead to the ability to produce routes with smaller total cost of all salesmen and highest cost of any single salesman. For the rest of the experimental studies, the weight setting $\omega_1 = 0.5, \omega_2 = 0.5$ is used.

7.6.2 Results for Two Objective Functions

Simulations were carried out to study the performance of the five algorithms applied on the MmTSP with different number of objective functions, salesmen, and cities. Figure 7.5 shows the evolved Pareto front of total travelling cost generated by the algorithms applied to the MmTSP with two objective functions, two salesmen, and 100 cities. It is observed that hREDA is able to produce a set of diverse solutions but it is slightly inferior in terms of proximity to the other algorithms. In general, hREDA has better convergence, in terms of IGD measurement, compared to other algorithms in the comparison as indicated in Figure 7.6(a). The indices of the IGD box-plot is shown in Table 7.3. REDA/D shows the worst performance even through its proximity is slightly better than or comparable to the other algorithms. This is due to the poor solution diversity. The domination-based algorithms (UMGA and NSGA-II) seems to outperform

dcomposition-based algorithms (REDA/D and MOEA/D) except hREDA. In terms of convergence speed (Figure 7.6(b)), all algorithms have a similar convergence speed at the early stages of evolution except UMGA. The convergence speed of REDA/D slowed down and outperformed by MOEA/D and NSGA-II thereafter. The poor diversity preservation in REDA/D caused the algorithm fail to further explore and exploit the promising search regions. When hybridization is carried, the diversity of REDA/D is enhanced by allowing exploitation of more neighbouring solutions. This hybridization rapidly improves the optimization performance in terms of IGD measurement. Thus, hREDA has the best performance in the later stages of evolution.

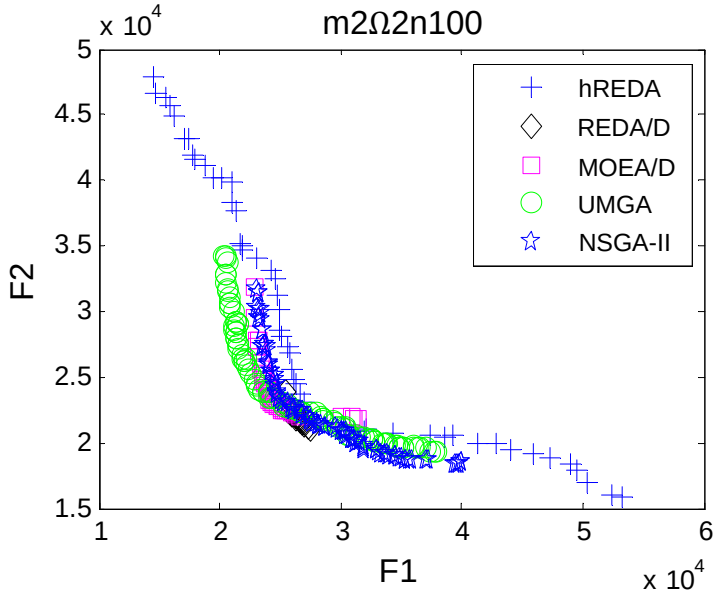
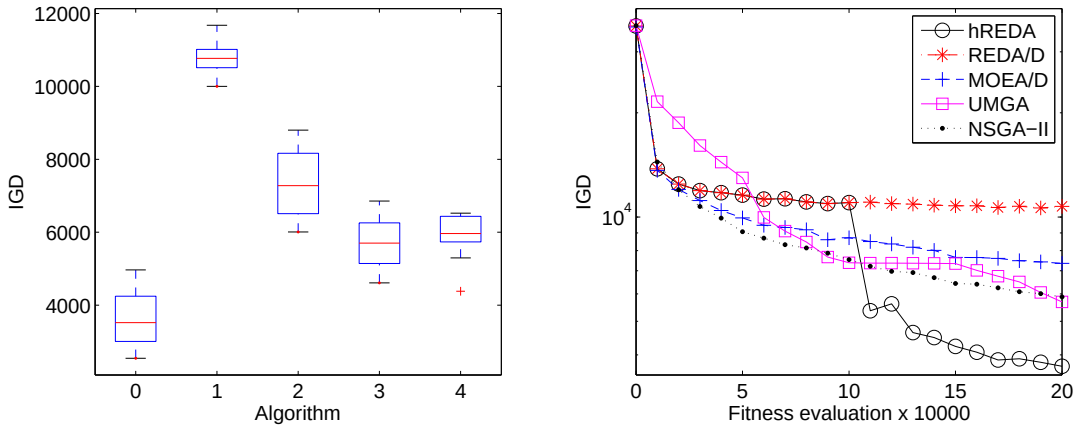


Figure 7.5: Evolved Pareto front of total travelling cost generated by the various algorithms applied to the MmTSP with two objective functions, two salesmen, and 100 cities

Table 7.3: Indices of the IGD box-plot

Index	0	1	2	3	4
Algorithm	hREDA	REDA/D	MOEA/D	UMGA	NSGA-II

The Pareto front for the MmTSP with 20 salesmen and 500 cities are shown in Figure 7.7. From the figure, it is observed that the decomposition-based algorithms (hREDA, REDA/D, and MOEA/D) achieve better Pareto front than the domination-based algorithms (UMGA and



(a) (b)

Figure 7.6: IGD and the convergence curve of total travelling cost generated by the various algorithms applied to the MmTSP with two objective functions, two salesmen, and 100 cities

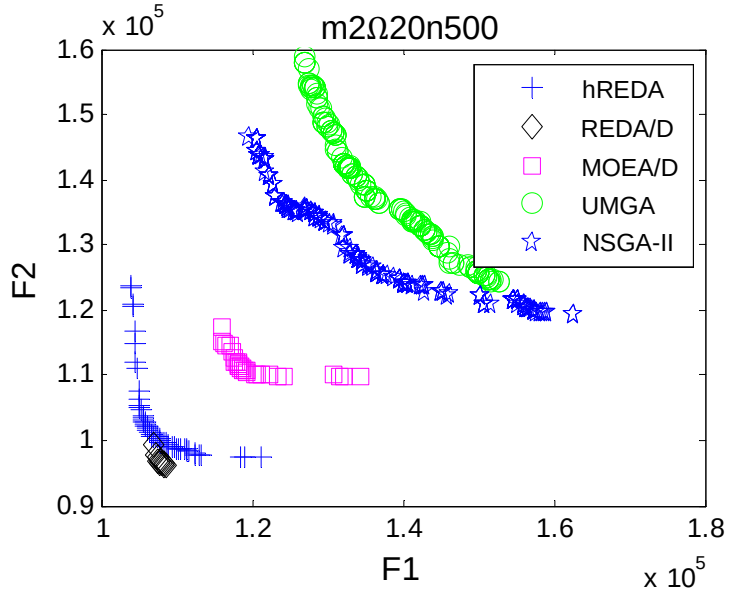
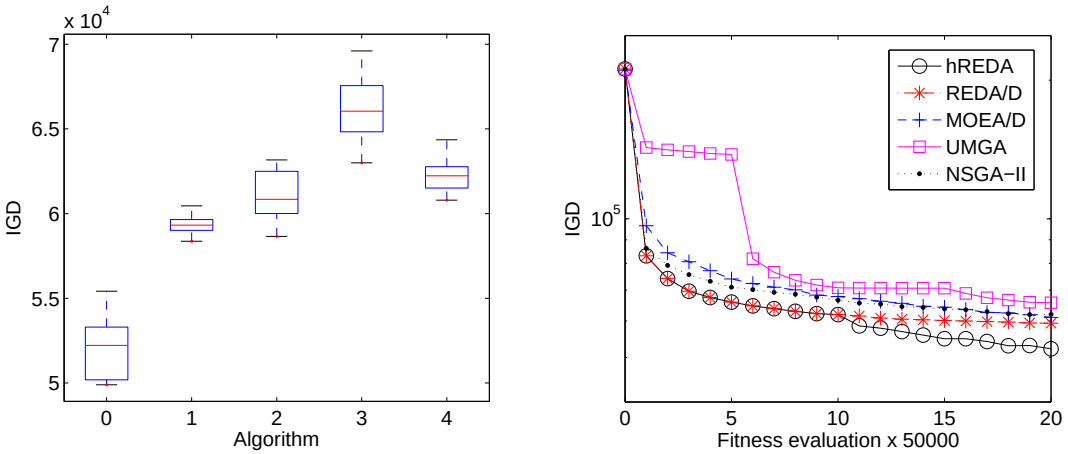


Figure 7.7: Evolved Pareto front of total travelling cost generated by the various algorithms applied to the MmTSP with two objective functions, 20 salesmen, and 500 cities

NSGA-II). For the decomposition-based algorithms, hREDA generates a better set of diverse solutions than REDA/D. However, the solutions generated by REDA/D have a better proximity than hREDA. In terms of IGD measurement as indicated in Figure 7.8(a), it is clear that hREDA has a better performance than REDA/D. This is because hREDA has a set of diverse solutions which is much better than REDA/D while REDA/D only slightly outperform hREDA in terms of proximity. Even through REDA/D has a poor solution diversity, its IGD value is better than



(a) (b)

Figure 7.8: IGD and the convergence curve of total travelling cost generated by the various algorithms applied to the MmTSP with two objective functions, 20 salesmen, and 500 cities

MOEA/D, UMGA, and NAGA-II, in which those algorithms has better solution diversity. This is because the proximity of REDA/D is much better than MOEA/D, UMGA, and NSGA-II. For convergence speed indicated in Figure 7.8(b), similar observation as the previous two test instances can be made. From this observation, it can be concluded that the decomposition-based algorithms scale well with the increase in the number of decision variables compared to the algorithms using the concept of domination. REDA/D uses global distribution of the parent solutions to guide the search process and is shown to have good proximity results, but poor solution diversity. Introducing local information into the evolutionary process, which helps the algorithm to further explore and exploit the search space, rectifies this limitation of REDA/D.

Table 7.4: IGD metric for total travelling cost for all salesmen of solutions obtained by various algorithms for the MmTSP with two objective functions, Ω salesmen, and n cities

Test Instance	hREDA	REDA/D	MOEA/D	UMGA	NSGA-II
m2Ω2n100	**3607±793**	10795±469	7349±952	5689±738	5881±651
m2Ω5n100	**4964±1041**	11253±785	8180±675	7376±776	6581±656
m2Ω10n100	8741±627	12332±501	8818±746	9428±840	**8288±671**
m2Ω2n300	**5779±757**	28319±648	26565±1093	24324±1474	25337±1455
m2Ω5n300	**8010±1805**	28898±805	27709±1125	26189±1480	27109±2244
m2Ω10n300	**17509±4213**	29828±1703	27667±761	32132±1523	27582±1757
m2Ω30n300	**31149±1823**	35198±1035	32554±1534	42952±1292	38103±2314
m2Ω2n500	**8358±2812**	57196±685	59346±1135	54436±2261	56002±1929
m2Ω5n500	**12114±2319**	58022±869	58811±1942	54424±1424	57047±2008
m2Ω10n500	**33680±13163**	58910±746	59622±1760	61855±24274	57599±1609
m2Ω20n500	**52195±1928**	59336±648	61076±1575	66323±2159	62250±1021
m2Ω50n500	**58870±2753**	65826±1456	69039±1610	82341±2635	77062±1298

For ease of visualization, the optimization results (mean ± standard deviation) for the dif-

ferent problem settings are presented in table form. In each table, the best result, in term of mean value, for each problem setting is highlighted in bold. Table 7.4 presents the total travelling cost for all salesmen of solutions generated by the algorithms for the MmTSP with two objective functions, Ω salesmen and n cities. The results show that hREDA generate the best solutions in most of the settings. This is caused by the usage of gradient information for local exploitation, which enhances the ability of the algorithm to search for more diverse solutions. From the table, it is also observed that the total travelling cost increases with the increase in the number of salesmen. This is because when more salesmen are involved, the task gets more difficult since the algorithms need to determine the route for each salesman while maintaining the minimum total travelling cost at the same time. Since all salesmen need to return to the home city and the final assigned city could be far from the depot, additional travelling cost may be incurred. For hREDA, the gradient information weakens with the increase in the number of salesmen, resulting in the algorithm not being able to exploit the information as effectively. However, its performance is still the best compared to the other four algorithms.

7.6.3 Results for Five Objective Functions

In order to understand the pairwise interactions among the five objective functions, 10 interaction fronts for three of the algorithms (hREDA, REDA/D, and MOEA/D) for the MmTSP with 10 salesmen and 300 cities are plotted in Figure 7.9. It is observed that hREDA has clearer interaction pattern than REDA/D and MOEA/D. Furthermore, hREDA has better proximity and spread in all the objective functions. Comparing the proximity of REDA/D and MOEA/D, it can be seen that they have similar performance in F1-F3, F2-F4, and F3-F4 interactions, REDA/D is inferior to MOEA/D in F1-F2 interaction, and REDA/D is superior to MOEA/D in the other remaining interactions. The corresponding IGD value and convergence trace are presented in Figure 7.10.

Table 7.5 shows the IGD metric for the total travelling cost of all salesmen of solutions

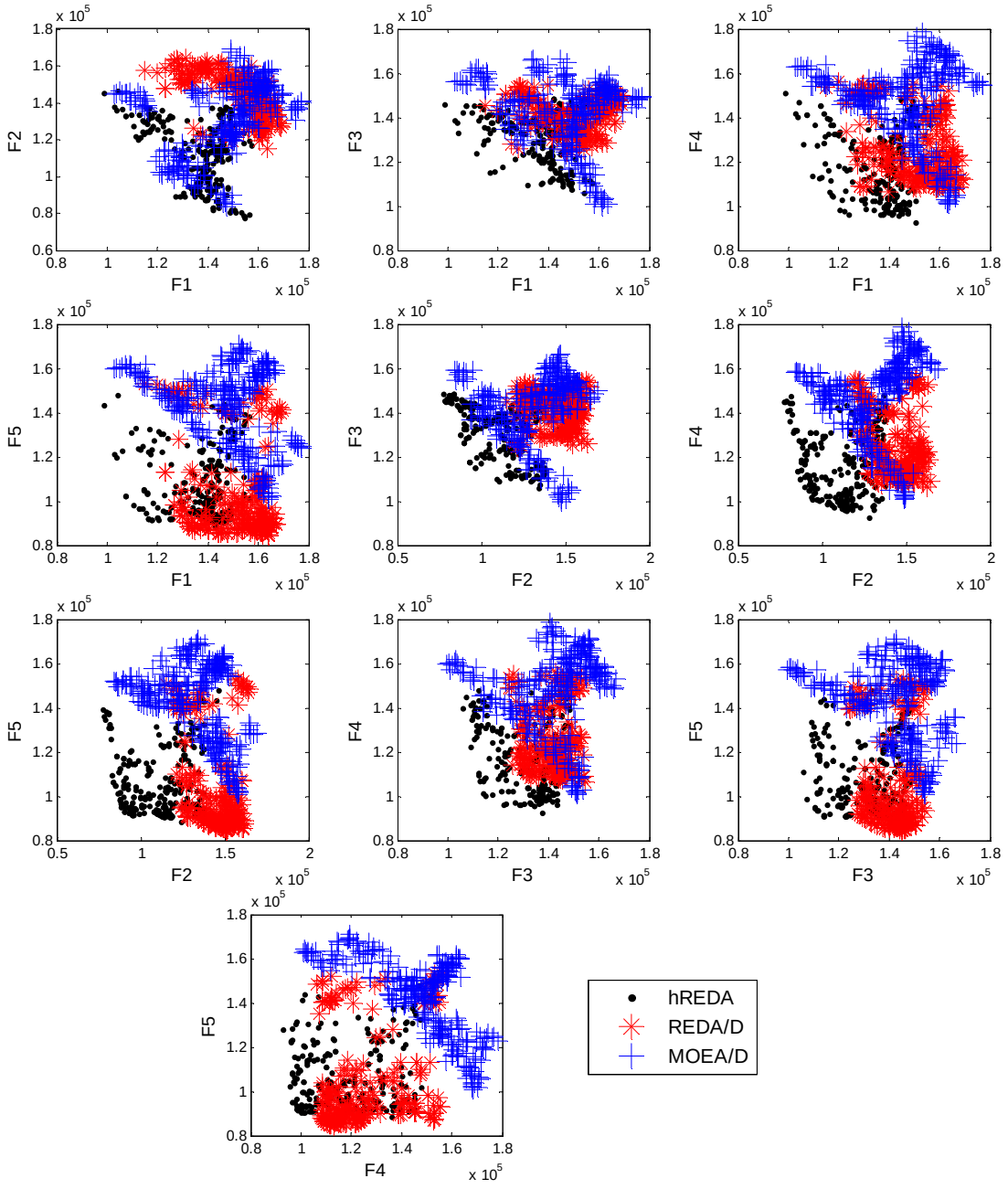
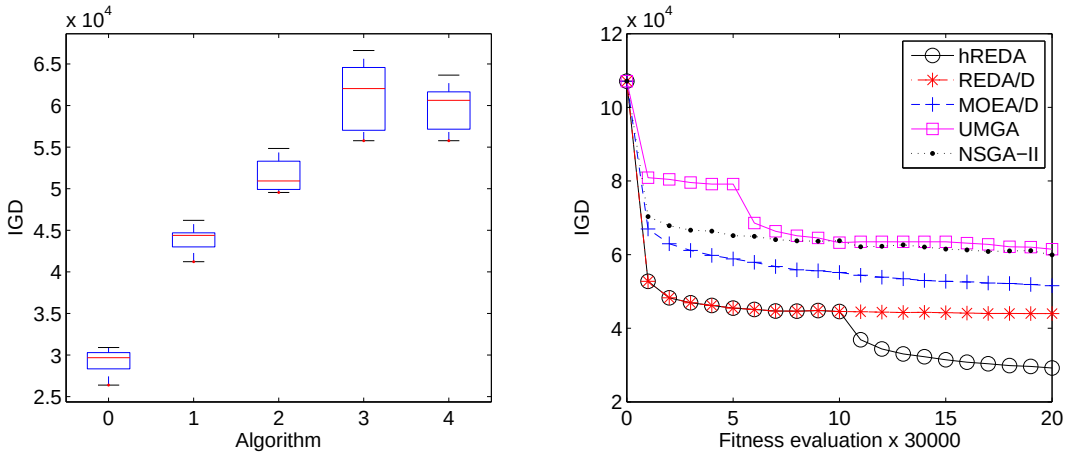


Figure 7.9: Evolved Pareto front of total travelling cost generated by the various algorithms applied to the MmTSP with five objective functions, 10 salesmen, and 300 cities

obtained by the algorithms for the MmTSP with five objective functions, different number of salesmen and cities. Generally, the performances of algorithms using the decomposition-based framework (hREDA, REDA/D, and MOEA/D) are superior to those of the algorithms based on the concept of domination (UMGA and NSGA-II) in most of the problem settings. The superiority of the decomposition-based algorithms is attributed to the aggregation principle used for fitness assignment. The tournament could be carried out by simply comparing the aggregated



(a) (b)

Figure 7.10: IGD and the convergence curve of total travelling cost generated by the various algorithms applied to the MmTSP with five objective functions, 10 salesmen, and 300 cities

Table 7.5: IGD metric for total travelling cost for all salesmen of solutions obtained by various algorithms for the MmTSP with five objective functions, Ω salesmen, and n cities

Test Instance	hREDA	REDA/D	MOEA/D	UMGA	NSGA-II
m5Ω2n100	**8306±689**	14045±885	8801±238	14170±549	13939±821
m5Ω5n100	**10007±1180**	13824±630	10623±677	17370±800	17364±861
m5Ω10n100	**12987±950**	15075±424	14098±900	23298±1347	23027±1118
m5Ω2n300	**16279±2203**	44048±1124	45971±1574	55795±2397	54134±1532
m5Ω5n300	**21666±3350**	44013±7616	46255±2229	56964±3343	56959±3782
m5Ω10n300	**29192±1492**	43987±1361	51566±1985	61475±4005	50017±2703
m5Ω30n300	**46069±1989**	52505±1111	73765±4077	95708±2438	93817±3599
m5Ω2n500	**24134±2105**	62362±785	95107±448	95930±2845	93413±3099
m5Ω5n500	**23939±5273**	62041±1812	96631±4331	95532±3843	94692±3449
m5Ω10n500	**37310±1217**	62770±1024	102910±2816	102940±3760	97700±4709
m5Ω20n500	**42030±2803**	63930±1404	108060±1912	107810±3999	100330±6145
m5Ω50n500	**69190±3972**	79290±2941	147620±4644	145400±4995	143650±5158

fitness values of solutions. Solutions with higher fitness values will always be selected to survive and reproduce. On the other hand, the concept of domination (NSGA-II and UMGA) requires that fitness be assigned to each solution based on their rank of domination. In many-objective problems, most of the solutions are non-dominated and are given lower ranks. This may prevent the tournament process from selecting promising solutions to survive.

7.7 Summary

This chapter proposed a hybrid EDA based on RBM for solving the MmTSP. The objective function of the MmTSP has been designed in the form of weighted sum of the total travelling

cost of all salesmen and the highest travelling cost of any single salesman. The proposed hybrid algorithm took into account the global information of the probability distribution of the cities and the local information in terms of trajectory of movements to perform the search. Furthermore, the utilization of the decomposition-based framework of multi-objective optimization succeeded in generating a set of promising tradeoff solutions in most of the instances of the MmTSP. This chapter also showed that the decomposition-based algorithms scale well in both the decision space and objective space. Comparative studies were carried out between the proposed algorithms and four state-of-the-art MOEAs, and the results indicated that REDA/D was able to achieve better results in larger problems and the hybridization with the EGS improved its performance.

Chapter 8

Hybrid Adaptive Evolutionary Algorithms for Multi-objective Optimization

Under the framework of evolutionary paradigms, many variations of evolutionary algorithms have been designed. Each of the algorithms performs well in certain cases and none of them are dominating one another. This study is based on the idea of synthesizing different evolutionary algorithms so as to complement the limitations of each algorithm. On top of this idea, this chapter proposes an adaptive mechanism that synthesizes a genetic algorithm, differential evolution, and estimation of distribution algorithm. The adaptive mechanism takes into account the ratio of the number of promising solutions generated by each optimizer in an early stage of evolutions so as to determine the proportion of the number of solutions to be produced by each optimizer in the next generation. Furthermore, the adaptive algorithm is also hybridized with the evolutionary gradient search to further enhance its search ability. The proposed hybrid adaptive algorithm is developed in the domination-based and decomposition-based frameworks of multi-objective optimization. An extensive experimental study is carried out to test the performances of the

proposed algorithms in 38 state-of-the-art benchmark test instances.

8.1 Background

In order to effectively solve an MOP, at least two issues need to be taken into consideration. The first issue is what algorithms are used to explore the search space and the second issue is what frameworks are used to find or maintain the multiple tradeoff Pareto optimal solutions. In the algorithmic issue, many multi-objective evolutionary algorithms (MOEAs) have been designed to solve MOPs. For example, MOEAs that use genetic algorithms (GAs) as the search technique are non-dominated sorting genetic algorithm II (NSGA-II) [32] and MOEA with decomposition (MOEA/D) [74], among others. MOEAs that use differential evolution (DE) as the search technique are Pareto differential evolution (PDE) [220], generalized differential evolution3 (GDE3) [221], and MOEA/D with DE [75], among others. Next, MOEAs that use estimation of distribution algorithms (EDAs) as the search approach are as described in Chapter 2. Each of the above-mentioned algorithms is efficient in solving certain MOPs and has their own strengths and weaknesses. Furthermore, no evidence indicates that one of the EAs is superior to the others. Thus, it is possible that the synthesis among the EAs can complement their weaknesses while maintaining their strengths.

In the framework's issue, at least three frameworks have been proposed to solve MOPs. They are the preference-based, domination-based, and decomposition-based frameworks. These frameworks have been briefly introduced in Section 1.1.4 and further discussed in Section 2.1. In the literature, many attempts have been devoted to studying the optimization performances of the algorithms in the domination-based and decomposition-based frameworks. However, the studies that aim to compare the optimization performances of both frameworks are considerably lacking.

This chapter has three main aims. First, a GA, DE, and EDA are synthesized in an adaptive manner. The adaptive feature takes into account the ratio of the number of promising solutions generated from each optimizer in an early stage of evolutions so as to determine the proportion

of the number of solutions to be produced by each optimizer in the next generation. The adaptive algorithm is also hybridized with a local search based on the evolutionary gradient search (EGS). Second, the hybrid adaptive algorithm is developed in the domination-based and decomposition-based frameworks of multi-objective optimization; thus, two hybrid adaptive algorithms are designed. Third, the optimization performance of the proposed algorithms is extensively studied in 38 state-of-the-art benchmark test instances which cover a wide range of characteristics such as deceptive, multimodality, different shapes of Pareto front, different number of decision variables, and different number of objective functions.

The rest of the chapter is organized as follows. Section 8.2 provides a literature review to the hybrid MOEAs. Section 8.3 presents the proposed algorithms in both the domination-based and decomposition-based frameworks of multi-objective optimization. Problem description and implementation of the simulations are given in Section 8.4. Simulation results are presented in Section 8.5. Finally, last section draws summary of this chapter.

8.2 Existing Studies

A hybrid algorithms is the combination between two algorithms in order to solve a problem. When a global search algorithm is hybridized with a local search algorithm, it is commonly known as memetic algorithm. Over the past decade, many hybrid algorithms have been devised. The two main issues in designing a hybrid algorithms are what algorithms to be combined and how to combine them.

The most common way to implement a local search in multi-objective optimization is to aggregate the multiple objective functions into a single-objective function using weighted sum methods. A common local search algorithm for single-objective optimization can then be directly implemented [18].

In [71], the authors introduced a multi-objective genetic local search algorithm (MOGLS). In MOGLS, the multiple objectives of an MOP is aggregated using a weighted sum approach.

The aggregated function of an offspring is used in performing the local search. In performing each local search, a vector of weights is randomly generated. Another attempt to use weighted sum approaches in performing local search for solving MOPs was suggested by Deb *et al.* [222]. The weighted sum approach is introduced to NSGA-II, in which the weight vector is determined based on each individual. The weight vector can also be determine by using simulated annealing as described in [223].

In [224], the authors proposed a memetic algorithm for Pareto archived evolution strategy algorithm (M-PAES). The evolution strategy is acted as a local search strategy while a population of candidate solutions is stored and recombination of the archived solutions (global search) is implemented.

In [138], Goh *et al.* developed a multi-objective evolutionary gradient search algorithm. The population-based approach of evolutionary computation is adapted into the evolutionary gradient search (EGS), such that the EGS can perform multi-directional search in the search space. Several approaches were used to derive the gradient information of multiple objectives of an MOP, such as a weighted sum approach with random weight vector [225], a goal programming technique [226], and a hypervolume indicator-based approach [227].

8.3 Proposed Hybrid Adaptive Mechanism

The fundamental idea of the proposed hybrid adaptive algorithms in this chapter is based on the assumption that combining the different EAs may complement the limitations of each optimizer while maintaining their strengths. On top of this idea, an adaptive feature, which determines the proportion of the number of solutions to be produced by each EA in a generation, is proposed. The adaptive feature is constructed in the domination-based and decomposition-based frameworks of multi-objective optimization. The constructed algorithms (global search) are then hybridized with the EGS, a local search, in order to enhance its exploitation ability. The hybrid adaptive algorithms are named as hybrid non-dominated evolutionary algorithm (hNSEA) and

hybrid MOEA/D (hMOEA/D).

%%Given a set of selected solutions that are stored in an archive (ψ)

1. Calculate the number of solutions in ψ that are generated from each EA, denoted as $D_g^{EA_i}$ where $i = 1, \ldots, M$, M is the number of EAs that are involved in the adaptive process
2. Calculate the adaptive proportion rate, denoted as $Ar_g^{EA\_i}$, for each EA
3. Check for the lower bound (l_bound) of the adaptive proportion rate

   For i = 1: M
   If $Ar_g^{EA\_i} < l\_bound$
   $Ar_g^{EA\_i} = l\_bound$
   End For

4. Normalize the adaptive proportion rate so that the sum of the adaptive proportion rates is equal to 1.0

   For i = 1: M
   $Ar_g^{EA\_i} = Ar_g^{EA\_i} / (\sum_{i=1}^{M} Ar_g^{EA\_i})$
   End For

Figure 8.1: Pseudo-code of the adaptive mechanism

The operations of the adaptive feature are as follows. Initially, each EA is given an equal chance to produce the initial solutions. After the reproduction processes, a number of promising solutions are selected and stored in an archive. Then, the proportion of the number of solutions to be generated by each optimizer in the next generation is calculated according to the proposed adaptive mechanism as illustrated in Figure 8.1. Let ψ as the solutions in an archive. First, calculate the number of solutions in ψ that are generated by each EA, denoted as $D_g^{EA_i}$, where $i \in \{1, ..., M\}$, M is the number of EAs that are involved in the adaptive process. In this chapter, three EAs are considered, including a GA, DE, and EDA. Thus, the number of solutions in ψ that are generated from each EA is denoted as $D_g^{EA_1} = D_g^{GA}$, $D_g^{EA_2} = D_g^{DE}$, and $D_g^{EA_3} = D_g^{EDA}$. Afterward, the adaptive proportion rate ($Ar_g^{EA_i}$) at generation g for each EA is calculated as follows:

$$Ar_g^{EA_i} = Ar_{g-1}^{EA_i} + \epsilon \times Pr_g^{EA_i} \tag{8.1}$$

$$Pr_g^{EA_i} = \frac{D_g^{EA_i}}{N}$$

where $Ar_g^{EA_i}$ is the adaptive proportion rate at generation g for i^{th} EA, ϵ is the learning rate, $Pr_g^{EA_i}$ is the current proportion rate, and N is the archive size or the number of solutions in an archive. A learning rate ($\epsilon < 0$) is incorporated to the updating rule in equation (8.1) in order to moderate the influences of the proportion of the number of selected solutions in generation g to the whole evolutionary process. This is because the optimizers that are able to generate a more number of promising solutions in the current generation may not be the superior optimizers in the next generation. Next, the adaptive proportion rate is set to a lower bound ($lbound$) value if it is lower than the $lbound$ as indicated in Step 3 of Figure 8.1. This is necessary since an optimizer may dominate other EAs and finally the adaptive proportion rate of this optimizer will become 1.0 while the adaptive proportion rate of other EAs will become 0.0. When this happens, all child solutions will only be generated by this optimizer till the end of the evolutionary process. Thus, it is necessary to set a lower bound to the adaptive proportion rate to guarantee that the problem would not exist. Since the summation of all the adaptive proportion rates should be equal to 1.0, the final adaptive proportion rates should be normalized especially when Step 3 is applied (Step 4). Afterward, a typical evolutionary process is continued.

The overall proposed hybrid evolutionary algorithm with non-dominated sorting approach (hNSEA) is presented in Figure 8.2. The algorithm starts with a random initialization of an initial population. All of the solutions in the population are evaluated to obtain their corresponding objective values. Next, all solutions are ranked according to the level of domination. For the solutions in the same rank, crowding distance is calculated. Then, select N promising solutions by using the binary tournament selection operator. Subsequently, the adaptive proportion rate for each EA is calculated as presented in Figure 8.1. In the reproduction stage, if GA is activated, the SBX and polynomial mutation are used to create an offspring. If DE is activated, the DE op-

Begin

1. **Initialization**: At generation $g = 0$, randomly generate N solutions to be the initial population, $Pop(g = 0)$
2. **Evaluation**: Evaluate each solution in the population

Do While ("Stopping criterion is not satisfied")

3. **Fitness assignment**: Apply the Pareto ranking and crowding distance over the population. Each solution consists of two values which represent its fitness, one is the rank of the domination and another is the level of crowded
4. **Selection**: Select N solutions using the binary tournament selection operator
5. **Adaptive**: Calculate the adaptive proportion rate for each EA
6. **Reproduction**: Build a probabilistic model $p_g\ (\boldsymbol{x})$ to represent the distribution of the solutions (required by REDA)

   For $\boldsymbol{i = 1}$: $\boldsymbol{N}$

   Generate a random value between 0 and 1 (u)

   If $u < Ar_g^{GA}$

   Generate an offspring using the SBX and polynomial mutation operators

   Else if $u \geq Ar_g^{GA}$ & $u < Ar_g^{GA} + Ar_g^{DE}$

   Generate an offspring using the DE and polynomial mutation operators

   Else

   Sample an offspring from $p_g\ (\boldsymbol{x})$

   End for
7. **Evaluation**: Calculate the objective values of all children solutions
8. **Archiving**: Store the parents and children solutions in an archive. Perform the Pareto ranking and crowding distance over the solutions in the archive
9. **Elitism**: Select N solutions with the lowest Pareto rank or highest crowding distance from the archive to form the new population
10. **Local Search**: Generate a random value between 0 and 1 (u)

    If $u < LS(local\ search\ rate)$

    For $\boldsymbol{i = 1}$: $\boldsymbol{N}$

    Generate another random value between 0 and 1 (u)

    If $u < K$ (Percentage of solutions undergoing local search)

    Perform EGS to generate an offspring

    End For

    Perform archiving and elitism to the parents and children solution to form the new population

End Do

End

Figure 8.2: Pseudo-code of the hybrid adaptive non-dominated sorting evolutionary algorithm (hNSEA)

erator and polynomial mutation are used to generate an offspring. Similarly, if EDA is activated, then an offspring is sampled from the constructed probabilistic model. After producing N child solutions, evaluation is performed to calculate their objective values. All of the parent and child solutions are stored in an archive. Elitism is performed to select N solutions with the lowest Pareto rank or highest crowding distance from the archive to form the new population. Next, the EGS is performed if it is activated. The description of the EGS can be referred to Section 7.2.2. The activation criterion is govern by a local search rate (LS). If a randomly generated number is smaller than LS, then local search is activated. In each activation, $K\%$ of the solutions in a

population is selected to generate neighbouring solutions using the EGS. All of the solutions generated from the EGS will also undergo archiving and elitism before forming a new population. A generation is terminated here. The evolutionary processes are continued until the maximum number of fitness evaluations is reached.

Begin

1. Initialization

a) Generate a set of uniformly distributed weight vectors ($\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$)

b) Calculate the Euclidean distance among the weight vectors. Determine the Q neighboring solutions ($B(i) = \{i_1, \ldots, i_Q\}, i \in [1, N]$) for each weight vectors according to the shortest Euclidean distance

c) At generation $g = 0$, randomly generate N solutions to be the initial population, $Pop(g = 0)$

d) Initialize reference point of the Tchebycheff approach ($\boldsymbol{z}^*$) by setting the value of $\boldsymbol{z}^*$ to be the lowest objective values of the solutions

Do while ("Stopping criterion is not satisfied")

Build a probabilistic model of the whole population $p_g(\boldsymbol{x})$ to represent the distribution of the solutions (required by REDA)

For $\boldsymbol{i} = \mathbf{1}$: $\boldsymbol{N}$

2. **Reproduction**: Generate a random value between 0 and 1 (u)

If $u < Ar_g^{GA}$

Randomly select two solutions from $B(i)$, and then generate an offspring using the SBX and polynomial mutation operators

Else if $u \geq Ar_g^{GA}$ & $u < Ar_g^{GA} + Ar_g^{DE}$

Randomly select three solutions from $B(i)$, and then generate an offspring using the DE and polynomial mutation operators

Else

Build another probabilistic model $p_g^{'}(\boldsymbol{x})$ which only considers the neighboring solutions

Generate an offspring: Generate a random value between 0 and 1(u)

For $\boldsymbol{j} = \mathbf{1}$: $\boldsymbol{n}$ (number of decision variables)

if ($u < 0.5$) sample from $p_g(\boldsymbol{x})$

else sample from $p_g^{'}(\boldsymbol{x})$

End For

3. **Evaluation**: Evaluate the generated offspring ($\boldsymbol{y}$) to obtain the corresponding objective values, $\boldsymbol{f}(\boldsymbol{y})$

4. **Update of $\boldsymbol{z}^*$:** For $j = 1, \ldots, m$, if $z_j^* > f_j(\boldsymbol{y})$, then set $z_j^* = f_j(\boldsymbol{y})$

5. **Fitness assignment**: Assign fitness to each solution (g^{te}) using Tchebycheff method

6. **Update Solution**: For $j \in B(i)$, if $g^{te}(\boldsymbol{y}|\boldsymbol{\lambda}^j, \boldsymbol{z}^*) \leq g^{te}(\boldsymbol{x}^j|\boldsymbol{\lambda}^j, \boldsymbol{z}^*)$, then set $\boldsymbol{x}^j = \boldsymbol{y}$ and $FV^j = \boldsymbol{f}(\boldsymbol{y})$

End For

7. **Local search:** Perform the EGS if it is activated (similar to hNSEA). Then, apply Steps 4-6 to update the reference point and the neighboring solutions

End Do

End

Figure 8.3: Pseudo-code of the hybrid MOEA/D (hMOEA/D)

The second proposed algorithm is the hybrid MOEA/D (hMOEA/D). hMOEA/D adapts the

decomposition-based algorithm suggested by Li and Zhang [75] and its process flow is illustrated in Figure 8.3. This study also uses Tchebycheff approach, as suggested by Li and Zhang [75], to transform an MOP into T scalar optimization subproblems. It is to be noted that $T = N$, which means that each subproblem is represented by a solution in the population. Initially, a set of uniformly distributed weight vectors ($\boldsymbol{\lambda}^i$, ...,$\boldsymbol{\lambda}^N$) are generated. The Euclidean distance between the weight vectors are calculated. Based on the shortest Euclidean distance, each weight vector is assigned Q neighbouring solutions (denoted as $B(i) = \{i_1, ..., i_Q\}, i \in [1, N]$). Next, an initial population is randomly generated. A reference point for the Tchebycheff approach ($\mathbf{z}^*$) is then initialized to be the lowest objective values of the solutions in the population. After that, the iterative process of the evolution starts. In the reproduction stage, if GA is activated, then two neighbouring solutions of solution i are randomly selected to undergo the SBX and polynomial mutation operators. If DE is activated, p_1 is set to be the solution i and then another two neighbouring solutions of solution i are randomly selected. A child solution is generated by the DE operator and polynomial mutation with a probability of p_m. If EDA is activated, then a random value between [0, 1] is generated in order to determine which probabilistic models are used to sample an allele of the child solutions. The first probabilistic model is constructed from all solutions in the population. The second probabilistic model is built from the neighbouring solutions of solution i. After a child solution was created, the solution is evaluated to obtain its objective values. Next, the reference point ($\mathbf{z}^*$) is updated. A fitness value is assigned to this solution using the Tchebycheff approach. Then, the neighbouring solutions are updated if the child solution is fitter than the neighbouring solutions. Finally, the EGS is performed if it is activated. The EGS is performed as discussed in Figure 8.2. A generation is completed here. The iterative process continues until the maximum number of fitness evaluations is reached.

Table 8.1: Parameter settings for experiments

Parameter	Setting
Population size, N	100 for problems with two objective functions, 300 for problems with three objective functions, and 500 for problems with five objective functions
Stopping criterion (number of fitness evaluations)	$500 \times N$
Number of runs	10
Lower bound ($lbound$) in the adaptive feature	0.1
Learning rate (ϵ) in the adaptive feature	0.1
Number of hidden units in REDA	5
Number of training epochs in REDA	2
Distribution index in the SBX and polynomial mutation (φ)	20
Mutation rate (p_m)	1/(Number of variables)
DE and crossover rate of the SBX	0.9
Local search rate (LS)	0.5
Percentage of solutions to undergo local search (K)	10% of the population size
Number of local neighbour (L) in the EGS	4
Number of neighbouring solutions in the decomposition-based algorithms (Q)	20

8.4 Problem Description and Implementation

In this chapter, eight algorithms were involved in the experimental studies. All of the algorithms were implemented in C++. The algorithms are NSGA-II with SBX operator (NSGA-II-SBX) [32], NSDE [92], MOEA/D with SBX operator (MOEA/D-SBX) [74], MOEA/D with DE (MOEA/D-DE) [75], NSREDA (used in the previous few chapters) , MOEA/D with REDA (MOEA/D-REDA), hNSEA, and hMOEA/D. The first four algorithms are the state-of-the-art algorithms of multi-objective optimization. NSREDA is the algorithm proposed in Chapter 4. In this chapter, we also adapt the NSREDA into the decomposition-based framework of multi-objective optimization(MOEA/D-REDA). hNSEA and hMOEA/D are the proposed hybrid adaptive algorithms. Thirty-one test instances with two or three objective functions (ZDT problems [101], DTLZ problems [102], UF problems [103], and WFG problems [104]) plus seven test instances with five objective functions (DTLZ problems) are used to test the optimization performance of the proposed algorithms. The detail description of the test problems can refer to Section 2.5. The performance metric of the inverted generational distance (IGD) is used to evaluate the optimization performances of the algorithms. The detail of the parameter settings is summarized in Table 8.1.

8.5 Results and Discussions

8.5.1 Comparison Results

Tables 8.2-8.4 shows the optimization results in terms of IGD measurement generated from the various algorithms. The parentheses next to the test problems refer to the number of decision variables (n) and objective functions (m) of the test problems. The average IGD values over 10 simulation runs are tabulated. The numbers inside the parentheses next to the IGD values refer to the ranking of the algorithms in a specific test instance. All rankings performed in this section, as summarized in Table 8.5, are based on the scores inside the parentheses.

ZDT problems, which possess two objective functions and a scalable number of decision variables, are a set of simple MOPs. Since all algorithms can easily solve the ZDT problems, we set the number of decision variables 10 times greater than its original setting. Thus, the ZDT problems have a large search space. The simulation results indicate that hNSEA has the best performance, followed by hMOEA/D. However, both of the algorithms fail to converge to the Pareto optimal front (PF) in ZDT4. This is because ZDT4 is an extremely multimodal problem which consists of many local optima. The results also show that EDA is able to generate a set of good results in ZDT problems, followed by GA and DE. In terms of the framework's issue, the domination-based algorithms generate better results than the decomposition-based algorithms. The ranking of the algorithms in ZDT problems is hNSEA, hMOEA/D, NSREDA, MOEA/D-REDA, NSGA-II-SBX, MOEA/D-SBX, MOEA/D-DE, and NSDE.

DTLZ problems consist of a scalable number of objective functions and decision variables. DTLZ1 and DTLZ3 are highly multimodal test problems. Due to the difficulties of both problems, we set the number of decision variables to 12. For the other DTLZ problems that are easier to solve, the number of decision variables is set to 120. The results indicate that, in DTLZ problems with three objective functions, hNSEA has the best results in DTLZ5, DTLZ6, and DTLZ7. The DTLZ5 and DTLZ6 have a degenerate PF, while DTLZ7 has a discontinuous PF.

Table 8.2: Results in terms of IGD measurement for ZDT, DTLZ, UF, WFG1, and WFG2 test problems

Algorithm	Test Instance (m, n)			
	ZDT1(2,300)	ZDT2(2,300)	ZDT3(2,300)	ZDT4(2,100)
NSGA-II-SBX	0.1979±0.0296(5)	0.3791±0.0564(5)	0.1585±0.0208(3)	25.320±2.0645(4)
MOEA/D-SBX	0.2997±0.0319(6)	0.4747±0.1405(6)	0.3141±0.0399(6)	29.053±3.6616(6)
NSDE	1.2686±0.1235(8)	2.5681±0.2241(8)	0.8484±0.1112(7)	27.687±6.2070(5)
MOEA/D-DE	1.0675±0.0353(7)	1.9065±0.1697(7)	0.9205±0.0442(8)	35.388±8.4534(7)
NSREDA	0.1497±0.0047(3)	0.2715±0.0084(3)	0.1696±0.0040(4)	18.217±1.5058(3)
MOEA/D-REDA	0.1895±0.0077(4)	0.3521±0.0189(4)	0.2016±0.0054(5)	16.253±1.9515(2)
hNSEA	**0.0148±0.0039(1)**	**0.0161±0.0008(1)**	**0.0126±0.0005(1)**	**7.3030±4.4287(1)**
hMOEA/D	0.0475±0.0062(2)	0.0683±0.0097(2)	0.0726±0.0088(2)	36.624±4.6961(8)
Algorithm	ZDT6(2,100)	DTLZ1(3,12)	DTLZ2(3,120)	DTLZ3(3,12)
NSGA-II-SBX	0.9203±0.0558(4)	**0.0146±0.0004(1)**	0.0458±0.0025(3)	0.0381±0.0025(3)
MOEA/D-SBX	0.6175±0.0391(2)	0.0173±0.0001(4)	0.0327±0.0007(2)	**0.0329±0.0004(1)**
NSDE	5.5331±0.2267(8)	1.2042±1.8231(6)	1.9567±0.2738(8)	4.0248±3.1192(3)
MOEA/D-DE	3.7106±0.2538(7)	0.1828±0.4989(5)	0.0659±0.0035(5)	0.0561±0.0591(5)
NSREDA	3.1699±0.0920(5)	11.006±1.8385(8)	0.3623±0.0638(7)	40.9702±7.9440(8)
MOEA/D-REDA	3.0413±0.0077(4)	8.4998±0.9996(7)	0.0776±0.0072(6)	28.875±7.5419(7)
hNSEA	**0.0103±0.0170(1)**	0.0169±0.0012(1)	0.0593±0.0020(4)	0.0456±0.0028(4)
hMOEA/D	0.6760±0.0381(3)	0.0172±0.0001(3)	**0.0320±0.0008(1)**	0.0331±0.0007(2)
Algorithm	DTLZ4(3,120)	DTLZ5(3,120)	DTLZ6(3,120)	DTLZ7(3,120)
NSGA-II-SBX	**0.0567±0.0020(1)**	0.0229±0.0008(2)	48.058±1.0496(6)	0.1008±0.0039(2)
MOEA/D-SBX	0.2183±0.1878(3)	0.0240±0.0001(4)	21.187±1.0420(4)	0.1742±0.0003(4)
NSDE	11.860±0.9960(7)	0.7055±0.1660(8)	57.367±3.6175(7)	3.0625±0.5678(8)
MOEA/D-DE	12.042±0.7354(8)	0.0274±0.0014(5)	3.1601±1.6770(3)	0.4761±0.0701(7)
NSREDA	1.6576±0.3273(6)	0.2927±0.0691(7)	71.850±4.3401(8)	0.4095±0.0254(6)
MOEA/D-REDA	1.0396±0.5263(5)	0.0377±0.0068(6)	0.0246±0.0001(2)	0.1679±0.0037(3)
hNSEA	0.2010±0.1280(2)	**0.0225±0.0003(1)**	**0.0235±0.0003(1)**	**0.0974±0.0044(1)**
hMOEA/D	0.2680±0.3012(4)	0.0239±0.0001(2)	27.410±4.8198(5)	0.1743±0.0006(5)
Algorithm	UF1(2,30)	UF2(2,30)	UF3(2,30)	UF4(2,30)
NSGA-II-SBX	0.1201±0.0244(6)	0.0479±0.0103(5)	0.2372±0.0404(5)	0.0538±0.0022(3)
MOEA/D-SBX	0.1258±0.0498(8)	0.0576±0.0297(6)	0.3094±0.0533(7)	0.0566±0.0047(4)
NSDE	0.0522±0.0128(3)	0.0450±0.0056(4)	0.1385±0.0318(3)	0.0730±0.0078(5)
MOEA/D-DE	0.0489±0.0290(2)	**0.0342±0.0237(1)**	**0.0745±0.0378(1)**	0.0824±0.0079(6)
NSREDA	0.1251±0.0395(7)	0.1023±0.0077(8)	0.3888±0.0766(8)	0.1347±0.0105(8)
MOEA/D-REDA	0.1181±0.0305(5)	0.0654±0.0075(7)	0.3088±0.0368(6)	0.0910±0.0073(7)
hNSEA	0.0549±0.0078(4)	0.0383±0.0042(2)	0.1292±0.0347(2)	**0.0507±0.0030(1)**
hMOEA/D	**0.0396±0.0113(1)**	0.0410±0.0217(3)	0.1812±0.0699(4)	0.0511±0.0030(2)
Algorithm	UF5(2,30)	UF6(2,30)	UF7(2,30)	UF8(3,30)
NSGA-II-SBX	**0.3087±0.1100(1)**	0.1462±0.0464(5)	0.1682±0.1338(5)	0.2203±0.0047(7)
MOEA/D-SBX	0.4436±0.1030(3)	0.1744±0.0549(8)	0.3222±0.1379(7)	0.1607±0.0379(4)
NSDE	0.8802±0.1721(8)	0.0442±0.0088(2)	0.0329±0.0090(4)	0.1478±0.0143(3)
MOEA/D-DE	0.6726±0.1360(7)	0.0466±0.0291(3)	0.0235±0.0063(2)	**0.0937±0.0082(1)**
NSREDA	0.5600±0.1262(6)	0.1719±0.0383(7)	0.3386±0.1624(8)	0.2971±0.0592(8)
MOEA/D-REDA	0.4930±0.1314(5)	0.1629±0.0666(6)	0.2988±0.1439(6)	0.1937±0.0123(5)
hNSEA	0.3163±0.0854(2)	**0.0437±0.0098(1)**	0.0249±0.0075(3)	0.1994±0.0363(6)
hMOEA/D	0.4871±0.1233(4)	0.1235±0.1592(4)	**0.0205±0.0063(1)**	0.1143±0.0103(2)
Algorithm	UF9(3,30)	UF10(3,30)	WFG1(2,30)	WFG2(2,30)
NSGA-II-SBX	0.1710±0.0423(5)	0.3274±0.0596(3)	1.3888±0.0891(8)	0.1017±0.0652(6)
MOEA/D-SBX	0.1211±0.0566(2)	0.3257±0.1810(2)	1.1816±0.1144(4)	0.1567±0.0374(8)
NSDE	0.1822±0.0672(6)	2.4853±0.2086(8)	1.2670±0.0118(7)	0.0306±0.0100(2)
MOEA/D-DE	**0.1058±0.0485(1)**	0.6108±0.0706(7)	1.2309±0.0033(6)	0.0500±0.0064(3)
NSREDA	0.4111±0.0418(8)	0.5053±0.0694(6)	1.1739±0.0057(3)	0.0983±0.0298(5)
MOEA/D-REDA	0.3661±0.0515(7)	0.4880±0.0780(5)	1.1832±0.0102(5)	0.1377±0.0572(7)
hNSEA	0.1444±0.0575(4)	**0.2629±0.0540(1)**	1.0036±0.0080(2)	**0.0270±0.0027(1)**
hMOEA/D	0.1326±0.0503(3)	0.3870±0.1280(4)	**0.9362±0.0096(1)**	0.0970±0.0626(4)

hMOEA/D has the best performances in DTLZ2, in which the problems has a spherical shape of PF. For problems with multi-modality (DTLZ1 and DTLZ3), the hybrid algorithms and the algorithms with SBX operator show better performances compared to the algorithms with DE and EDA. This finding is identical to the results obtained in [9] which concluded that EDAs are

Table 8.3: Results in terms of IGD measurement for WFG3-WFG9 and DTLZ1-DTLZ5 with five objective test problems

Algorithm	Test Instance (m, n)			
	WFG3(2,30)	WFG4(2,30)	WFG5(2,30)	WFG6(2,30)
NSGA-II-SBX	0.0250±0.0014(2)	0.0184±0.0008(3)	0.0671±0.0010(3)	0.0481±0.0041(3)
MOEA/D-SBX	0.0299±0.0033(5)	**0.0155±0.0005(1)**	**0.0665±0.0007(1)**	0.0447±0.0070(2)
NSDE	**0.0245±0.0011(1)**	0.0994±0.0036(8)	0.0733±0.0011(6)	0.0819±0.0204(8)
MOEA/D-DE	0.0292±0.0026(4)	0.0791±0.0103(7)	0.0673±0.0002(4)	0.0818±0.0182(7)
NSREDA	0.0709±0.0197(7)	0.0325±0.0038(5)	0.0771±0.0014(8)	0.0611±0.0059(4)
MOEA/D-REDA	0.0959±0.0247(8)	0.0506±0.0058(6)	0.0756±0.0014(7)	0.0693±0.0125(5)
hNSEA	0.0260±0.0010(3)	0.0219±0.0012(4)	0.0700±0.0010(5)	0.0815±0.0162(6)
hMOEA/D	0.0320±0.0012(6)	0.0164±0.0007(2)	0.0668±0.0001(2)	**0.0362±0.0090(2)**
Algorithm	WFG7(2,30)	WFG8(2,30)	WFG9(2,30)	DTLZ1(5,12)
NSGA-II-SBX	0.0172±0.0007(3)	0.0803±0.0038(3)	0.0200±0.0019(3)	22.503±13.459(6)
MOEA/D-SBX	**0.0141±0.0001(1)**	0.0766±0.0054(2)	0.0177±0.0013(2)	0.2176±0.0025(2)
NSDE	0.0332±0.0018(6)	0.1272±0.0121(7)	0.0335±0.0008(4)	62.370±21.402(7)
MOEA/D-DE	0.0180±0.0008(4)	0.1119±0.0112(5)	0.0348±0.0193(5)	2600.94±162.91(8)
NSREDA	0.0421±0.0081(7)	0.1393±0.0106(8)	0.0551±0.0158(8)	4.9827±8.1962(4)
MOEA/D-REDA	0.0457±0.0215(8)	0.1261±0.0104(6)	0.0384±0.0193(6)	3.7365±1.5907(3)
hNSEA	0.0189±0.0009(5)	0.0856±0.0029(4)	0.0432±0.0097(7)	12.359±5.1947(5)
hMOEA/D	**0.0141±0.0001(1)**	**0.0741±0.0023(1)**	**0.0165±0.0006(1)**	0.2145±0.0002(1)
Algorithm	DTLZ2(5,120)	DTLZ3(5,12)	DTLZ4(5,120)	DTLZ5(5,120)
NSGA-II-SBX	1.5922±0.0627(5)	75.415±18.419(6)	6.0306±0.8802(4)	4.5888±0.5141(6)
MOEA/D-SBX	1.0260±0.0501(2)	1.0020±0.0315(3)	**1.1211±0.1238(1)**	0.9331±0.0027(2)
NSDE	3.9636±3.1544(6)	217.49±30.572(7)	21.193±0.6283(7)	1.3406±0.6291(5)
MOEA/D-DE	1.0535±0.0044(4)	960.33±373.29(8)	9.2750±0.6970(5)	0.9375±0.0035(4)
NSREDA	10.4467±0.8754(8)	12.157±5.7649(4)	25.163±0.6490(8)	9.1865±0.5106(8)
MOEA/D-REDA	**0.9940±0.0449(1)**	17.991±4.4488(5)	4.5550±1.0588(3)	0.9369±0.0030(3)
hNSEA	6.6723±1.2337(7)	**0.9197±0.1300(1)**	17.972±1.1783(6)	7.0585±0.6700(7)
hMOEA/D	1.0302±0.0479(3)	0.9335±0.0050(2)	1.7985±0.3771(2)	**0.9307±0.0000(1)**

Table 8.4: Results in terms of IGD measurement for DTLZ6 and DTLZ7 with five objective test problems

Algorithm	Test Instance (m, n)	
	DTLZ6(5,120)	DTLZ7(5,120)
NSGA-II-SBX	96.699±0.7107(8)	4.8243±0.1021(5)
MOEA/D-SBX	15.582±0.4147(5)	4.2058±0.0095(3)
NSDE	80.677±7.0645(7)	9.0199±0.9137(7)
MOEA/D-DE	10.3338±2.4412(4)	11.9862±0.4779(8)
NSREDA	20.2494±6.3474(6)	**3.6190±0.0245(1)**
MOEA/D-REDA	0.9297±0.0013(2)	4.3186±0.1403(4)
hNSEA	**0.9008±0.0000(1)**	4.1183±0.2459(2)
hMOEA/D	0.9307±0.0000(3)	6.1212±0.6330(6)

easily trapped in local optima. DTLZ4 has a bias landscape when mapping from the decision space to the objective space. NSGA-II-SBX yields the best result and followed by hNSEA. The algorithms with DE operator show the worse IGD value. In terms of the framework's issue, there is no clear results indicating that one of the frameworks is better than another. The ranking of the algorithms in DTLZ problems with three objective functions is hNSEA, NSGA-II-SBX, MOEA/D-SBX, hMOEA/D, MOEA/D-REDA, MOEA/D-DE, NSREDA, and NSDE.

In DTLZ problems with five objective functions, the results indicate that hMOEA/D and MOEAD-SBX have the best results. Even though hNSEA is able to obtain the best results in

Table 8.5: Ranking of the algorithms in various test problems

Rank	Test Problem(Score)		
	ZDT	DTLZ-3	UF
1	hNSEA(5)	hNSEA(15)	hNSEA(26)
2	hMOEA/D(17)	NSGA-II-SBX(18)	hMOEA/D(28)
3	NSREDA(19)	MOEA/D-SBX(22)	MOEA/D-DE(31)
4	MOEA/D-REDA(20)	hMOEA/D(23)(20)	NSGA-II-SBX(45)
5	NSGA-II-SBX(21)	MOEA/D-REDA(36)	NSDE(46)
6	MOEA/D-SBX(26)	MOEA/D-DE(38)	MOEA/D-SBX(51)
7	NSDE(36)	NSREDA(42)	MOEA/D-REDA(59)
8	MOEA/D-DE(36)	NSDE(50)	NSREDA(74)
Rank	WFG	DTLZ-5	Overall
1	hMOEA/D(19)	hMOEA/D(18)	hMOEA/D(105)
2	MOEA/D-SBX(26)	MOEA/D-SBX(18)	hNSEA(112)
3	NSGA-II-SBX(34)	MOEA/D-REDA(21)	MOEA/D-SBX(143)
4	hNSEA(37)	hNSEA(29)	NSGA-II-SBX(158)
5	MOEA/D-DE(45)	NSREDA(39)	MOEA/D-DE(191)
6	NSDE(49)	NSGA-II-SBX(40)	MOEA/D-REDA(194)
7	NSREDA(55)	MOEA/D-DE(41)	NSDE(227)
8	MOEA/D-REDA(58)	NSDE(46)	NSREDA(229)

DTLZ3 and DTLZ6, its performances in other DTLZ problems are poor, except DTLZ7. The decomposition-based algorithms show better IGD results than the domination-based algorithms. The ability of the decomposition-based algorithms in solving problems with many objective functions has been discussed in Section 2.1 that the decomposition-based algorithms can differentiate the superiority of the solutions by using the aggregated fitness values while the decomposition-based algorithms need to determine the domination behaviours between the solutions before the superiority of the solutions is determined, where the domination behaviours is weakened with the increase in the number of objective functions. The experimental results also show that the performances of EDA are superior to GA and DE. This finding is consistent with the results reported in Chapter 3. The ranking of the algorithms in DTLZ problems with five objective functions is hMOEA/D, MOEA/D-SBX, MOEA/D-REDA, hNSEA, NSREDA, NSGA-II-SBX, MOEA/D-DE, and NSDE.

UF problems can be regarded as a set of MOPs with complicated shapes of PS. It is a set of difficult MOPs that were proposed for the CEC 2009 competition. We preserve the number of decision variables to be 30 even though they are scalable. UF1-UF7 consist of two objective functions and UF8-UF10 possess three objective functions. The simulation results show that MOEA/D-DE obtains the best IGD values in four UF test problems, hNSEA obtaining three best

results, hMOEA/D obtaining two best results, and NSGA-II-SBX obtaining one best results. The MOEAs with EDA give inferior solutions to most of the UF test problems. This may be cause by the fact that the EDA fails to construct a probabilistic model to represent the complicated distribution of the solutions in the decision space. Besides, the ability of the algorithms in generating a set of diverse solutions is critical in addressing the UF problems [75]. In [89], the researchers claimed that EDA is particularly weak in generating a set of diverse solutions since only global information of the probability distribution is used. Thus, REDA performs weakly in UF test problems. In terms of the framework's issue, there is no clear superiority in the performances for both the domination-based and decomposition-based algorithms. The ranking of the algorithms in UF problems is hNSEA, hMOEA/D, MOEA/D-DE, NSGA-II-SBX, NSDE, MOEA/D-SBX, MOEA/D-REDA, and NSREDA.

WFG problems are another set of difficult MOPs that involve various types of transformations. The problems consist of a scalable number of objective functions and decision variables. In this chapter, two objective functions and 30 decision variables are applied. The decision vector consists of two position parameters and 28 distance parameters. The simulation results show that hMOEA/D generates a set of solutions with the best IGD values in five WFG problems. The performances of the algorithms with GA are better than the algorithms with DE and EDA. In terms of the framework's issue, the decomposition-based algorithms outperform the domination-based algorithms slightly. The ranking of the algorithms in WFG problems is hMOEA/D, MOEA/D-SBX, NSGA-II-SBX, hNSEA, MOEA/D-DE, NSDE, NSREDA, and MOEA/D-REDA.

Overall, hNSEA obtains the best IGD results in 14 test problems followed by hMOEAD with the best IGD results in 10 test problems. These findings demonstrate that the proposed hybrid adaptive mechanism improves the optimization performance of an individual optimizer. However, it also happens that the hybrid adaptive algorithms generate the worse IGD values, as indicate in ZDT4. To sum up, the hybrid adaptive mechanism proposed in this chapter succeeds in complementing the limitations of an individual EA and in maintaining their search abilities in

most of the test instances. In terms of individual EA, the performances of GA are superior in DTLZ problems with three and five objective functions and WFG problems. Its performances in ZDT and UF problems are average. The performances of DE are superior in UF problems, average in WFG problems and inferior in ZDT and DTLZ problems with three and five objective functions. The performances of EDA are superior in ZDT, average in DTLZ problems with three and five objective functions and inferior in UF and WFG problems. In terms of the framework's issue, the domination-based algorithms are superior to the decomposition-based algorithms in ZDT problems and inferior in the other test problems. The overall ranking of the algorithms in all the test problems is hMOEA/D, hNSEA, MOEA/D-SBX, NSGA-II-SBX, MOEA/D-DE, MOEA/D-REDA, NSDE, and NSREDA. The ranking is summarized in Table 8.5.

8.5.2 Effects of Local Search on Optimization Performance

Figure 8.4 shows the effects of different local search rate on optimization performance. Six cases are shown in order to highlight different observations. Overall, the performance of hNSEA is better in most of the test problems when local search is activated (local search rate, LS = 0 means local search is deactivated). The box-plot for ZDT1 shows this observation. However, it may also happen that hNSEA has a better performance when $LS = 0$ as shown in DTLZ2 and WFG6. In UF3, $LS = 0.5$ gives the best performance compared to other settings. An interesting observation can be seen in UF6 that the activation of local search reduce the number of outliers. Figure 8.5 shows the effects of the number of solutions undergoes local search on optimization performance. In ZDT1, 25% of solutions in the population undergo local search give the best performance compared to other settings. In DTLZ2, the performance deteriorates when higher numbers of solutions are exposed to local search. Figure WFG2 shows the reverse observation as DTLZ2, in which the performance is better when more number of solutions are undergone local search.

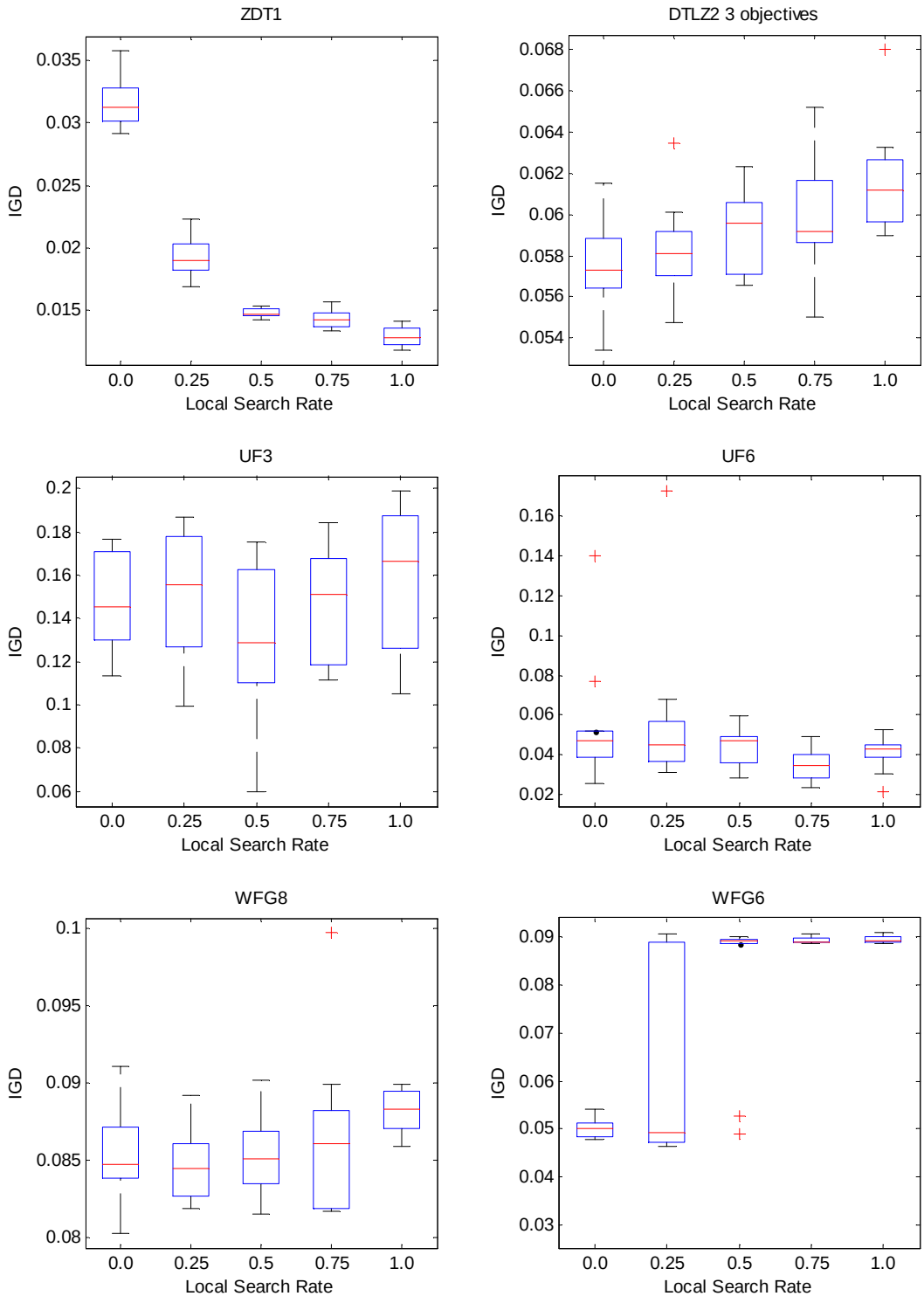


Figure 8.4: Effects of local search rate on optimization performance

8.5.3 Effects of Adaptive Feature on Optimization Performance

Three EAs are considered in the hybrid adaptive mechanism. The adaptively activation plot of the different EAs is shown in Figure 8.6. All plots in the figure show that GA will dominate

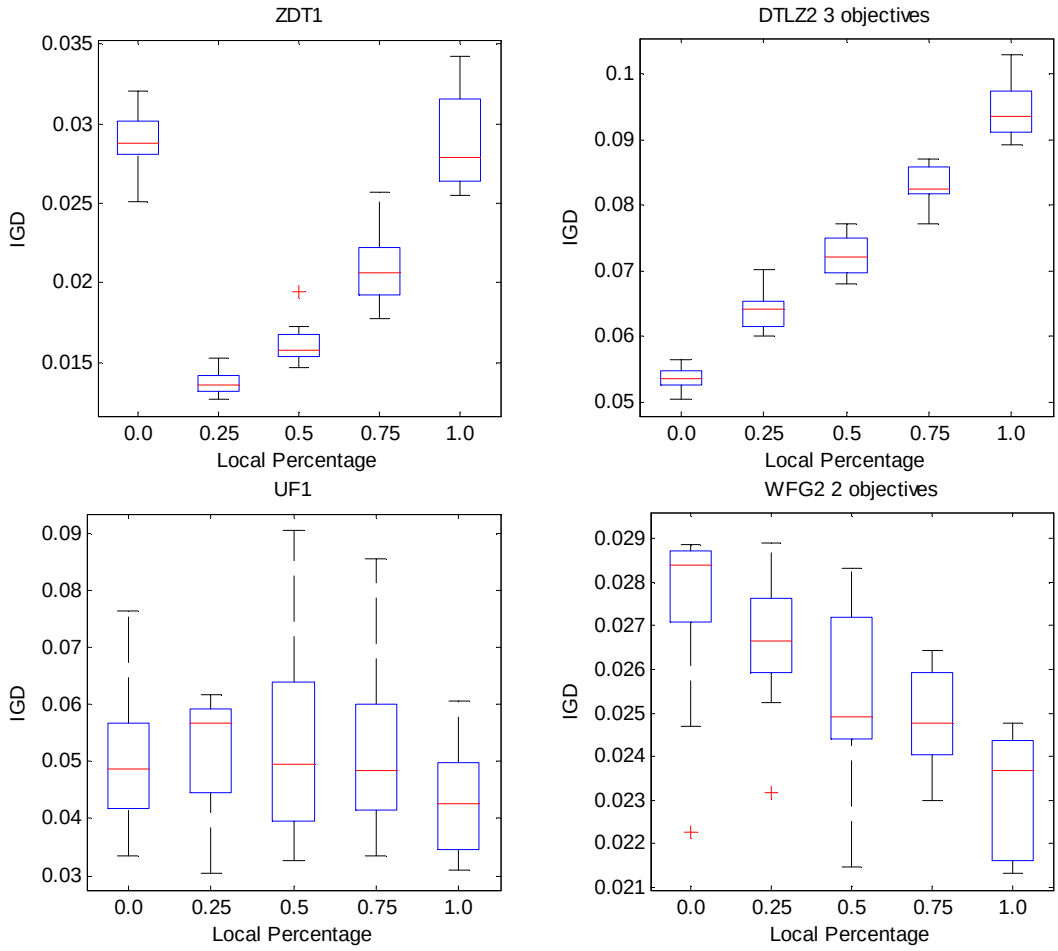


Figure 8.5: Effects of the percentage of local search on optimization performance

the search at the later stages of evolution. The activation of DE and EDA in an early stage of evolution is problem dependent.

There are two parameters introduced to the adaptive mechanism: lower bound ($lbound$) and learning rate ϵ. $lbound$ is important in the case that there exists an algorithm dominates other in an early stage of evolution but that algorithm will lead the search towards local optima. Figure 8.7 shows the optimization performance in IGD measurement with regards to different settings of $lbound$. Only results for four test problems with different performance are shown. $lbound = 0$ means that no lower bound is set. The figure indicates that the settings of lower bound give superior optimization performance in some test problem (e.g. ZDT4), inferior in some test problems (e.g. ZDT1), and have comparable performance in other test problems (e.g.

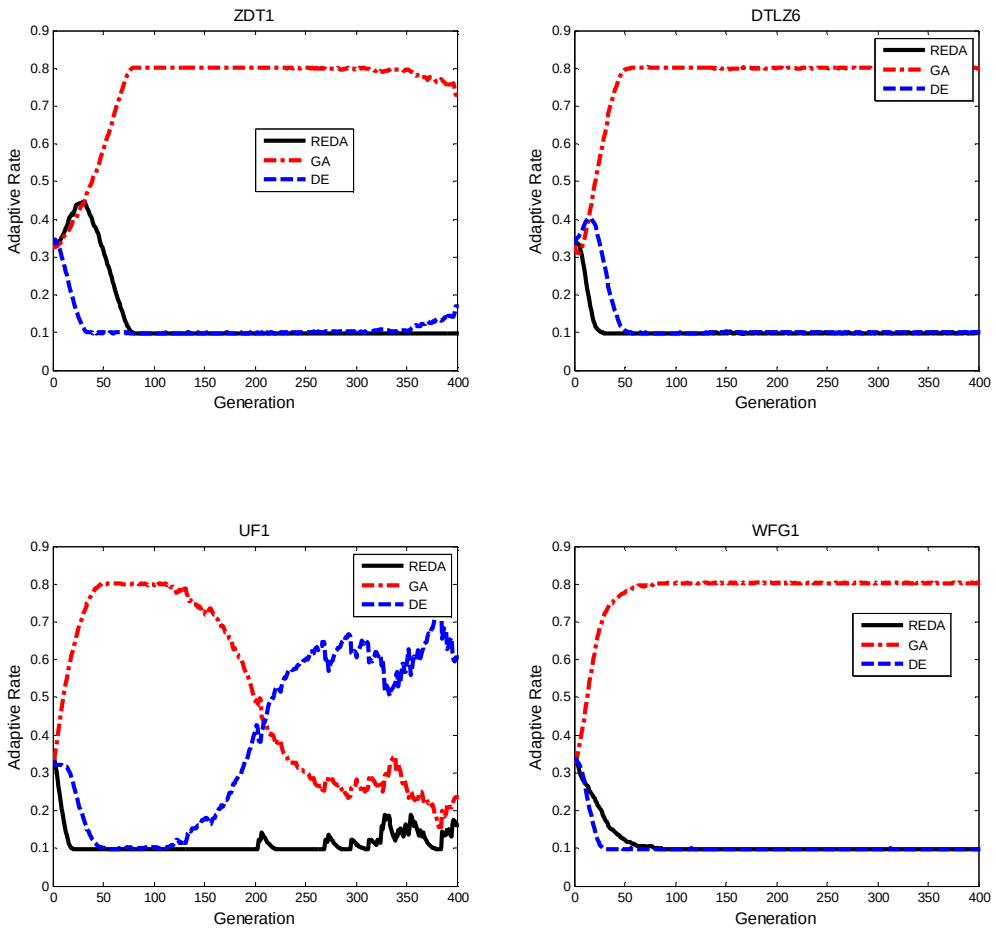


Figure 8.6: Adaptive activation of different EAs

DTLZ5 with 3 objectives). $lbound = 0.33$ means that the adaptivity is deactivated. The IGD results for $lbound = 0.33$ are inferior in most of the test problems (e.g. ZDT1, ZDT4, and DTLZ5 with 3 objectives) and superior in UF8.

ϵ is introduced in the proposed adaptive mechanism to moderate the influences of the proportion of the number of selected solutions in generation g to the whole evolutionary process. This is because the optimizers that are able to generate a more number of promising solutions in the current generation may not be the superior optimizers in the next generation. Figure 8.8 shows the effects of learning rate on optimization performance. Only results for four test problems with different performance are shown. In ZDT1, a smaller value of ϵ give better optimization results. In DTLZ2 with 3 objectives, a higher value of ϵ give better optimization results. In UF8 and WFG2, no clear trend is observed of the settings of ϵ.

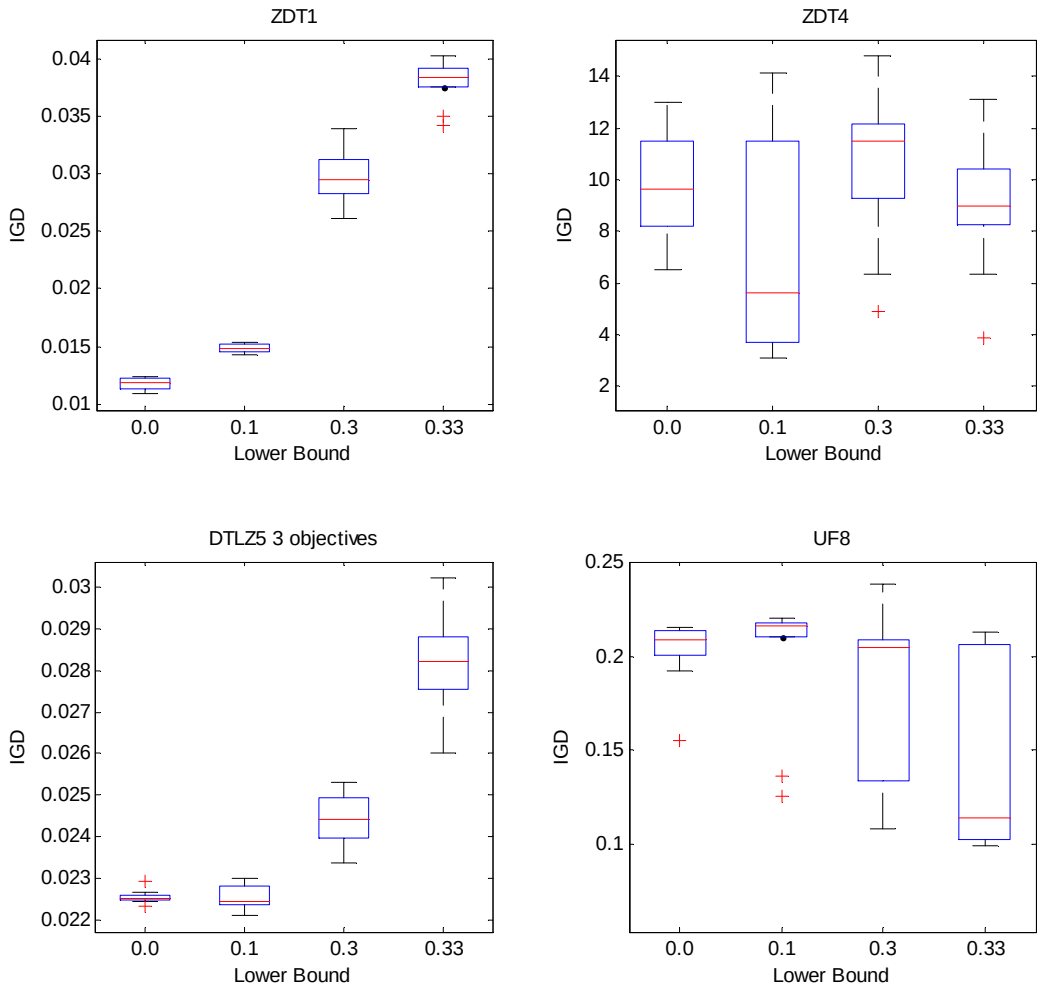


Figure 8.7: Effects of lower bound on optimization performance

8.6 Summary

This chapter proposed an adaptive mechanism to synthesize a GA, DE, and EDA for multi-objective optimization. The adaptive mechanism takes into account the ratio of the number of promising solutions generated by each optimizer in an early stage of evolutions so as to determine the proportion of the number of solutions to be produced by each optimizer in the next generation. The search ability of the adaptive algorithm was further enhanced by hybridizing it with a local search based on the EGS. The hybrid adaptive algorithm was constructed in the domination-based and decomposition-based frameworks of multi-objective optimization. The performances of the hybrid adaptive algorithms, together with other six state-of-the-art MOEAs, were tested in 38 benchmark test instances. The simulation results showed that the proposed algorithms outper-

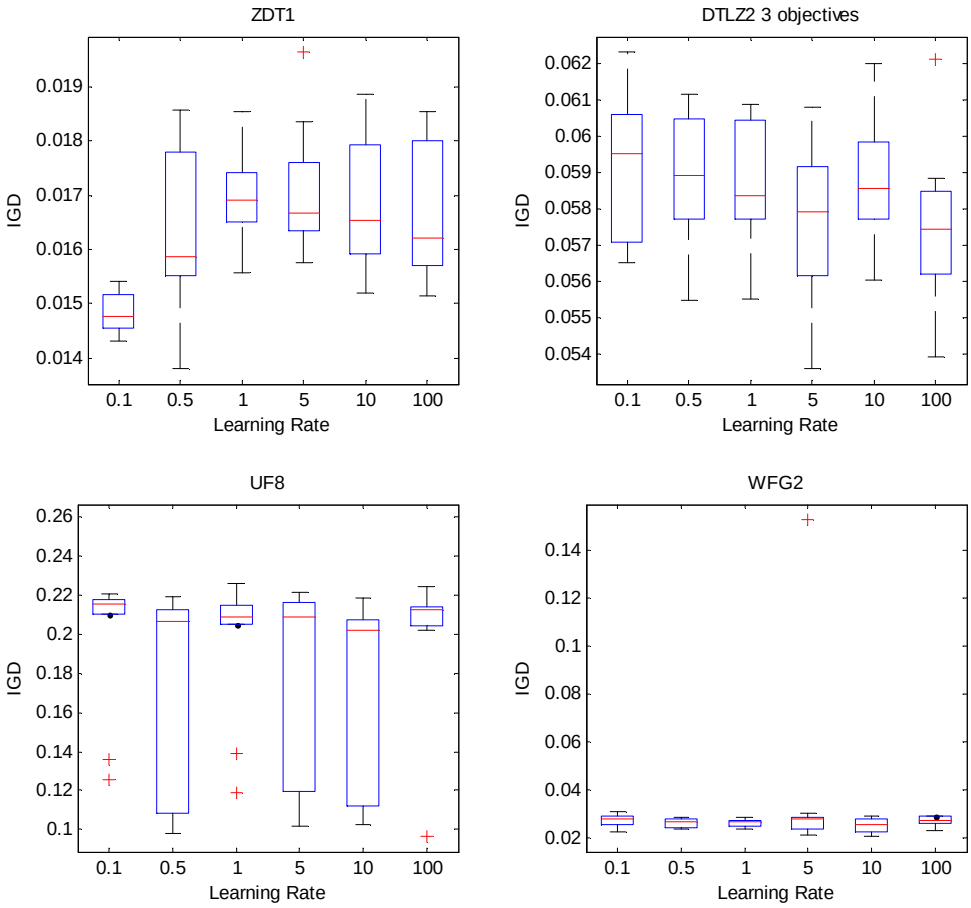


Figure 8.8: Effects of learning rate on optimization performance

form the other six algorithms. The results also demonstrated that the optimization performances of the decomposition-based algorithms are clearly superior to the domination-based algorithms in the test problems with five objective functions.

Chapter 9

Conclusions

9.1 Conclusions

The primary aim of this thesis is to propose an optimization algorithm employing the synergy between evolutionary computation and probabilistic graphical models to solve a variety of multi-objective optimization problems (MOPs) effectively. In chapter 3, an algorithm in the evolutionary multi-objective framework using restricted Boltzmann machine-based estimation of distribution algorithm (REDA) has been designed. The simulation results have showed that REDA scales well with an increase in the number of decision variables as well as the number of objective functions compared to the other three state-of-the-art algorithms. These results tally with another study which investigates the optimization performance of EDAs in decomposable scalability issues [88]. The capability of EDAs, particularly REDA, in solving high-dimensional MOPs can be attributed to the utilization of probability distribution of promising solutions as well as the incorporation of linkage information into the search process. This is different comparing to algorithms that using genetic operators in which the offspring are generated by randomly performing crossover and mutation mechanisms. The simulation results have also showed that REDA is able to converge to global optima in a smaller number of fitness evaluations. These results suggest that REDA is suitable for use in solving complex and difficult MOPs. The capability of

REDA in solving high-dimensional MOPs in limited computational resources is of crucial importance especially when dealing with real-world optimization problems where a fitness evaluation is computationally expensive. However, REDA suffers several limitations. REDA requires a large amount of computational resources comparing to NSGA-II and MOUMDA. Furthermore, REDA is easily trapped at local optima and it has a poor capability to escape from those optima. Since no location and neighboring solutions are taken into consideration, REDA is particularly weak in generating a set of diverse solutions especially when the search space is biased, multimodal, and complex.

In chapter 4, an extensive investigation has been carried out to examine the behaviours of the training, modelling, and sampling issues of an RBM in the perspective of evolution. The investigation results have showed that an extensive training of RBM is unnecessary, a solution has a smaller energy level if it is located in the regions modelled by an RBM, and a solution has a higher energy level if it is located in the regions outside of the modelled topology. The later two results indicate that focusing the search on the modelled topologies may enhance the exploitation ability of an algorithm, while focusing the search outside the modelled topologies may help an algorithm to explore the search space. These observations have motivated the proposal of the energy-based sampling mechanism in order to enhance the search capability of REDA. The simulation results have showed that the energy-based sampling mechanism with inverse exponential selection scheme has significantly improved the optimization performance of REDA in both static and epistatic MOPs. These results suggest that a higher selection pressure should be given to solutions with lower energy levels while a lower selection pressure should be preserved for solutions with higher energy levels. However, the energy-based REDA may cause premature convergence especially in a highly biased test problem due to a strong exploitation in a particular search region. Besides, it incurs extra computational time and is not well in escaping the local optima.

In chapter 5, an attempt has been carried out to study the optimization performance of

REDA in solving MOPs with noisy objective functions. Different noise levels with normal distribution have been considered. A likelihood correction operator has been devised to adjust the probabilistic model so that it is less affected by the solutions with large selection errors. Furthermore, a hybrid algorithm between REDA and PSO has been proposed. The simulation results have showed that REDA is more robust compared to NSGA-II in noisy environments and the likelihood correction operator slightly improves the optimization performance of REDA. The results have also indicated that the hybrid algorithm enhances the search capability of REDA. One of the possibilities is that the particles may now move out of the regions modelled by the probabilistic model, providing extra solutions which are unexplored by the model. The possibility of other hybridizations between REDA and DE, and REDA and GA has also been considered. It has found that all hybridizations improve the optimization performance in noisy MOPs. This result suggests that the search capability of EDAs (global search algorithms) can be enhanced when hybridizing with any other global search algorithms. This study shows for the first time that the hybrid EDA is able to generate a set of approximate Pareto optimal solutions in noisy MOPs. However, this study only considers gaussian noise. Thus, REDA is not a generic optimization approach in noisy environments. Furthermore, REDA fails in approximating the Pareto optimal front when a high level of noise is added.

In Chapter 6, a study has been carried out to implement REDA in solving the multi-objective travelling salesman problem (MOTSP). Several adaptations and problem specific operators have been designed, namely integer-number representation, permutation operator, and problem specific local search operator. The results have showed that the solutions generated by REDA have poor solution diversity but good solution proximity. On the other hand, GA has the opposite results in which the generated solutions has poor solution proximity but good solution diversity. The observation that REDA fails to generate a set of diverse solutions tallies with the results obtained from the previous three chapters. This is due to the nature of the probabilistic modelling in REDA which only models the promising solutions in certain search topologies. These

results have motivated the proposal of the synthesized algorithm between REDA and GA in an alternating manner. The alternating manner approach has showed promising improvement, in which the synthesized algorithm had the best optimization performance in terms of proximity and diversity. This shows for the first time that the diversity of an EDA can be enhanced by designing a synthesized algorithm in an alternating manner, which is different from the previous chapters in which the diversity of EDAs is improved through clustering or hybridization with other search algorithms. However, the final obtained results are not optimal. Furthermore, no preference knowledge are taken into consideration during optimization.

In chapter 7, a study has been conducted to examine the capability of REDA in solving the multi-objective multiple travelling salesmen problem (MmTSP). A new formulation of the fitness function for the MmTSP and a new hybrid algorithm between REDA and a local search in the decomposition-based framework have been designed. It has found that REDA was able to generate a set of tradeoff solutions with better IGD results than the other algorithms in the comparison. These findings are most likely a result of the positive effect of RBM as well as the incorporation of gradient information into the search process. The capability of hybrid REDA in generating a sequence of cities in approximate optimal ordering implies that hybrid REDA can be used to deal with any scheduling or logistic problems in which the problems by nature involve permutation. This study is of considerable importance since it sheds new light on the possibility of implementing EDAs to solve permutation-based problems. Moreover, it has also found that the proposed formulation for the MmTSP eliminated the problem of uneven assignation of tasks to each salesman. This is because the formulation takes weighted sum measurements between the lowest travelling cost of all salesmen and the highest travelling cost of any single salesman in order to balance the workloads assigned to each salesman. A set of alternative solutions representing the tradeoff among the various conflicting objectives coupled with an easily adjustable level of task assignation in REDA provide multiple choices of solutions to a logistic manager. However, the approximated Pareto front is not optimal due to a slow convergence after a certain

number of fitness evaluations.

In chapter 8, an attempt has been conducted to combine three EAs, including EDA, GA, and DE, in an adaptive manner. The adaptive algorithm has also hybridized with a local search algorithm. The simulation results have showed that the hybrid adaptive algorithms in both the domination-based and decomposition-based frameworks of multi-objective optimization yield the best results in most of the test problems compared to the other six state-of-the-art algorithms. This is most probably due to the positive effect of the adaptive hybridization between several optimization algorithms. Even through three EAs are considered in this thesis, the proposed adaptive mechanism is a generic approach which can combine any number of search algorithms. The hybrid adaptive algorithms are holistic and generic optimization algorithms that can be applied to solve a variety of MOPs effectively. However, the algorithms may fail in approximating the Pareto optimal front when the search space is highly biased, multi-modal, and deceptive.

9.2 Future Work

Considering that the implementation of EDAs in multi-objective optimization is noticeably lacking in contemporary research, more insightful studies are needed in future. The sampling mechanism of EDAs applied in this thesis only took into account marginal probability information. This probability information may restrict the sampling mechanism of REDA in producing new solutions especially when the decision variables of objective functions are highly correlated. To address this problem, the probability information which takes into account the conditional probability distribution of solutions could be implemented in future work. Using the conditional probability distribution, the sampling mechanism may consider the explicit probability of existence of cardinalities of a decision variable with regard to cardinalities of other decision variables.

While this study has demonstrated the efficiency of REDA in solving MOPs, the ability of REDA in modelling the correlations between the decision variables has not been proven or investigated. Moreover, the modelling technique of REDA is recognized to be inefficient for

modelling large and complex data in machine learning fields. To address this issue, future studies could utilize a deep learning network, which has been proven to be able to effectively model complex data, and to capture the probability distribution of solutions.

In chapters 6 and 7, REDA has been implemented to solve permutation-based MOPs, with the restriction in the travelling salesman problem (TSP). Even though TSP is a benchmark scheduling problem which is closely related to other real-world scheduling and logistic problems, the application of REDA in the TSP and real-world scheduling problems may consist of several differences. These differences may arise in the solution representation, constraints, and level of conflict between objective functions, among others. Thus, another possible avenue of future work is to adapt EDAs in general, or REDA in particular to solve real-world permutation-based problems, for example, exam timetabling problems, berth allocation problems, gene sequencing problems, and network routine problems, among others. When dealing with real-world permutation-based problems, a priori knowledge of the problems can be taken into consideration, through a local search operator or a specific enhancement operator, in an optimization process. The incorporation of this knowledge into an optimization process may enhance the search capability of an algorithm and thus may provide a better solution quality.

This thesis has implemented EDA in general, or REDA in particular to study global continuous MOPs, scalable MOPs, epistatic MOPs, noisy MOPs, and permutation-based MOPs. Many other characteristics and issues of MOPs, which are out of the scope of this thesis, could be studied in future. These issues include parallelism, interactivity, multi-modality, uncertainty, and framework. One of the main limitations of EDAs is that EDAs require large computational resources in a single evolutionary process. Thus, parallelism of EDAs is a promising area which aims to tackle that limitation. EDAs can also be designed to interact with users (interactivity issue) especially in deciding the likeliness of users towards certain choices. In addition, multi-modality is always a challenging issue in optimization. Studies of EDAs in this aspect are considerably lacking.

Many cost functions of real-world problems are uncertain in nature. The cost functions may be subject to noise in the objective space, which is commonly known as noisy MOPs. Only normally distributed noise has been studied in this thesis. The effects of other types of noise could be studied. The objective functions may also be subject to noise in the decision space, and it is commonly known as robust MOPs. For some cost functions, the Pareto optimal solutions may change over time. It is commonly known as dynamic MOPs. There has been no attempt to study the robustness and dynamic issues of MOPs using EDAs. For the framework issue, most of the EDAs have been developed in the domination-based framework; two attempts in this thesis have developed EDAs in the decomposition-based framework. However, there has been no study that develops EDAs in the preference-based framework. Thus, this is also a promising research direction that can be explored in future work.

Bibliography

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*. Chichester John Wiley and Sons, 2001.

[2] K. C. Tan, E. F. Khor, and T. H. Lee, *Multiobjective evolutionary algorithms and applications*. Springer, 2005.

[3] S. Huang and Y. Zhu, "NSGA-II based grid task scheduling with multi-QoS constraint," in *Proceedings of the Third International Conference on Genetic and Evolutionary Computing*, pp. 306–308, 2009.

[4] B. Malekmohammadi, B. Zahraie, and R. Kerachian, "Ranking solutions of multi-objective reservoir operation optimization models using multi-criteria decision analysis," *Expert Systems with Applications*, pp. 7851–7863, 2011.

[5] Y. Wu, L. Xu, and J. Xue, "Improved multiobjective particle swarm optimization for environmental/economic dispatch problem in power system," in *Proceedings of the Second International Conference on Advances in Swarm Intelligence*, pp. 49–56, 2011.

[6] B. Barán, C. von Lücken, and A. Sotelo, "Multi-objective pump scheduling optimisation using evolutionary strategies," *Advances in Engineering Software*, vol. 36, no. 1, pp. 39–47, 2005.

[7] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms. A new tool for evolutionary computation*. Kluwer Academic Publishers, 2001.

[8] M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective estimation of distribution algorithm," in *Scalable Optimization via Probabilistic Modeling*, pp. 223–248, Springer, 2006.

[9] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: Methods and applications*. Ph.D. Dissertation, Swiss Federal Institute of Technology, Zurich, 1999.

[10] P. Dasgupta, P. P. Chakrabarti, and S. C. Desarkar, *Multiobjective heuristic search: An introduction to intelligent search methods for multicriteria optimization*. Springer, 1999.

[11] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithms: Analyzing the state-of-the-art," *Evolutionary Computation*, vol. 8, no. 2, pp. 125–147, 2000.

[12] C. K. Goh and K. C. Tan, *Evolutionary multi-objective optimization in uncertain environments: Issues and algorithms*. Springer, 2009.

[13] C. Y. Cheong, *Evolutionary multi-objective optimization in scheduling problems*. Ph.D. Dissertation, National University of Singapore, 2009.

[14] D. A. Coley, *An introduction to genetic algorithms for scientists and engineers*. World Scientific Publishing.

[15] R. Sivaraj and T. Ravichandran, "A review of selection methods in genetic algorithm," *International Journal of Engineering Science and Technology*, vol. 3, no. 5, pp. 3792–3797, 2011.

[16] K. Deb and H.-G. Beyer, "Self-adaptive genetic algorithms with simulated binary crossover," *Evolutionary Computation*, vol. 9, no. 2, pp. 197–221, 2001.

[17] R. Dawkins, *The selfish gene*. Oxford University Press, New York, 1976.

[18] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary algorithms for solving multi-Objective problems*. Springer, 2006.

[19] C. A. Coello Coello, "Evolutionary multi-objective optimization: A historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28–36, 2006.

[20] H. Mühlenbein and G. Paass, "From recombination of genes to the estimation of distributions i. binary parameters," in *Proceedings of the Fourth International Conference on Parallel Problem Solving from Nature*, pp. 178–187, 1996.

[21] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 41–63, 2008.

[22] Y. Y. Haimes, L. S. Lasdon, and D. A. Wismer, "On a bicriterion formulation of the problems of integrated system identification and system optimization," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 1, no. 3, pp. 296–297, 1971.

[23] K. Mietten, *Nonlinear multiobjective optimization*. Boston: Kluwer, 1999.

[24] L. Rachmawati and D. Srinivasan, "Multiobjective evolutionary algorithm with controllable focus on the knees of the Pareto front," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 810–824, 2009.

[25] L. Thiele, K. Miettinen, P. J. Korhonen, and J. M. Luque, "A preference-based evolutionary algorithm for multi-objective optimization," *Evolutionary Computation*, vol. 17, no. 3, pp. 411–436, 2009.

[26] A. Zhou, B. Qu, H. Li, S. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, pp. 32–49, 2011.

[27] D. E. Goldberd, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[28] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multi-objective optimization: formulation, discussion and generalization," in *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 416–423, 1993.

[29] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[30] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, p. 257271, 1999.

[31] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," Technical Report 103, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland, 2001.

[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[33] D. Brockhoff and E. Zitzler, "Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods," in *IEEE Congress on Evolutionary Computation*, pp. 2086–2093, 2007.

[34] J. Bader and E. Zitzler, "Robustness in hypervolume-based multiobjective search," Technical Report: TIK 317, Computer Engineering and Networks Laboratory, ETH Zurich, 2010.

[35] B. B. Li and L. Wang, "A hybrid quantum-inspired genetic algorithm for multiobjective flow shop scheduling," *IEEE Transactions on Systems, Man and Cybernetics: Part B (Cybernetics)*, vol. 37, no. 3, pp. 576–591, 2007.

[36] A. Elhossini, S. Areibi, and R. Dony, "Strength Pareto particle swarm optimization and hybrid EA-PSO for multi-objective optimization," *Evolutionary Computation*, vol. 18, no. 1, pp. 127–156, 2010.

[37] W. F. Leong and G. G. Yen, "PSO-based multiobjective optimization with dynamic population size and adaptive local archives," *IEEE Transactions on Systems, Man and Cybernetics: Part B (Cybernetics)*, vol. 38, no. 5, pp. 1270–1293, 2008.

[38] A. Lara, G. Sanchez, C. A. Coello Coello, and O. Schutze, "HCS: A new local search strategy for memetic multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 112–132, 2010.

[39] C. K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 103–127, 2009.

[40] C. K. Goh, K. C. Tan, D. S. Liu, and S. C. Chiam, "A competitive and cooperative coevolutionary approach to multi-objective particle swarm optimization algorithm design," *European Journal of Operational Research*, vol. 202, no. 1, pp. 42–54, 2010.

[41] K. C. Tan, C. K. Goh, Y. J. Yang, and T. H. Lee, "Evolving better population distribution and exploration in evolutionary multi-objective optimization," *European Journal of Operational Research*, vol. 171, no. 2, pp. 463–495, 2006.

[42] V. L. Huang, S. Z. Zhao, R. Mallipeddi, and P. N. Suganthan, "Multi-objective optimization using self-adaptive differential evolution algorithm," in *IEEE Congress on Evolutionary Computation*, pp. 190–194, 2009.

[43] M. L. Wong, "Parallel multi-objective evolutionary algorithms on graphics processing units," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pp. 2515–2522, 2009.

[44] A. J. Nebro and J. J. Durillo, "A study of the parallelization of the multi-objective metaheuristic MOEA/D," in *Fourth International Conference on Learning and Intelligent Optimization*, pp. 303–317, 2010.

[45] F. Pettersson, N. Chakraborti, and H. Saxén, "A genetic algorithms based multiobjective neural net applied to noisy blast furnace data," *Applied Soft Computing*, vol. 7, no. 1, pp. 387–397, 2007.

[46] C. K. Goh and K. C. Tan, "An investigation on noisy environments in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 3, pp. 354–381, 2007.

[47] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: Test cases, approximations, and applications," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004.

[48] Z. Bingul, "Adaptive genetic algorithms applied to dynamic multiobjective problems," *Applied Soft Computing*, vol. 7, no. 3, pp. 791–799, 2007.

[49] Z. Cai and Y. Wang, "A multiobjective optimization-based evolutionary algorithm for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 658–675, 2006.

[50] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema, "Constraint handling in multiobjective evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 514–525, 2009.

[51] K. C. Tan, C. Y. Cheong, and C. K. Goh, "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation," *European Journal of Operational Research*, vol. 177, no. 2, pp. 813–839, 2007.

[52] R. Alcala, "A fast and scalable multiobjective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 4, pp. 666–681, 2011.

[53] D. Brockhoff and E. Zitzler, "Objective reduction in evolutionary multiobjective optimization: Theory and applications," *Evolutinoary Computation*, vol. 17, no. 2, pp. 135–166, 2009.

[54] H. K. Singh, A. Isaacs, and T. Ray, "A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 539–556, 2011.

[55] J. Yao, N. Kharma, and P. Grogono, "Bi-objective multipopulation genetic algorithm for multimodal function optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 80–102, 2010.

[56] K. Deb and A. Saha, "Finding multiple solutions for multimodal optimization problems using a multi-objective evolutionary approach," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pp. 447–454, 2010.

[57] W. Gong and Z. Cai, "An improved multiobjective differential evolution based on Pareto-adaptive epsilon-dominance and orthogonal design," *European Journal of Operational Research*, vol. 198, no. 2, pp. 576–601, 2009.

[58] B. Y. Qu and P. N. Suganthan, "Multi-objective differential evolution with diversity enhancement," *Journal of Zhejiang University Science A*, vol. 11, pp. 538–543, 2010.

[59] D. S. Liu, K. C. Tan, S. Y. Huang, C. K. Goh, and W. K. Ho, "On solving multiobjective bin packing problems using evolutionary particle swarm optimization," *European Journal of Operational Research*, vol. 190, no. 2, pp. 357–382, 2008.

[60] S. Agrawal, B. K. Panigrahi, and M. K. Tiwari, "Multiobjective particle swarm algorithm with fuzzy clustering for eletrical power dispatch," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 5, pp. 529–541, 2008.

[61] C. Igel, N. Hansen, and S. Roth, "Covariance matrix adaptation for multi-objective optimization," *Evolutionary Computation*, vol. 15, no. 1, pp. 1–28, 2007.

[62] W. Dong and X. Yao, "Unified eigen analysis on multivariate Gaussian based estimaion of distribution algorithms," *Information Sciences*, vol. 178, no. 15, pp. 3000–3023, 2008.

[63] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Annals of Operational Research*, vol. 131, no. 1-4, pp. 79–99, 2004.

[64] C. García-Martínez, O. Cordón, and F. Herrera, "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," *European Journal of Operational Research*, vol. 127, no. 1, pp. 116–148, 2007.

[65] M. Gong, H. Jiao, H. Du, and L. Bo, "Multiobjective immune algorithm with nondominated neighbor-based selection," *Evolutionary Computation*, vol. 16, no. 2, pp. 225–255, 2008.

[66] Z. H. Hu, "A multiobjective immune algorithm based on a multiple-affinity model," *European Journal of Operational Research*, vol. 202, no. 1, pp. 60–72, 2010.

[67] Y. Zhang and P. Rockett, "A generic optimising feature extraction method using multiobjective genetic programming," *Applied Soft Computing*, vol. 11, no. 1, pp. 1087–1097, 2011.

[68] A. Zafra, E. L. G. Galindo, and S. Ventura, "Multiple instance learning with multiple objective genetic programming for web mining," *Applied Soft Computing*, vol. 11, no. 1, pp. 93–102, 2011.

[69] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using Pareto archived evolutionary strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[70] S. Huband, P. Hingston, L. White, and L. Barone, "An evolution strategy with probabilistic mutation for multi-objective optimization," in *IEEE Congress on Evolutionary Computation*, pp. 2284–2291, 2003.

[71] H. Ishibuchi, T. Yoshida, and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transaction on Systems, Man, and Cybernetics: Part C (Applications and Reviews)*, vol. 28, no. 3, pp. 204–223, 1998.

[72] A. Jaszkiewicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 402–412, 2002.

[73] L. Paquete and T. Stützle, "A two-phase local search for the biobjective traveling salesman problem," in *Proceedings of the Second International Conference on Evolutionary Multicriterion Optimization*, pp. 479–493, 2003.

[74] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[75] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.

[76] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *IEEE Congress on Evolutionary Computation*, pp. 203–208, 2009.

[77] P. Palmers, T. McConnaghy, M. Steyaert, and G. Gielen, "Massively multi-topology sizing of analog integrated circuits," in *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 706–711, 2009.

[78] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Simultaneous use of different scalarizing functions in MOEA/D," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pp. 519–526, 2010.

[79] P. C. Chang, S. H. Che, Q. Zhang, and J. L. Lin, "MOEA/D for flowshop scheduling problems," in *IEEE Congress on Evolutionary Computation*, pp. 1433–1438, 2008.

[80] S. Pal, B. Qu, S. Das, and P. N. Suganthan, "Optimal synthesis of linear antenna arrays with multi-objective differential evolution," *Progress in Electromagnetics Research B*, vol. 21, pp. 87–111, 2010.

[81] T. J. Yuen and R. Raml, "Comparison of computational efficiency of MOEA/D and NSGA-II for passive vehicle suspension optimization," in *Proceedings of 24th European Conference on Modelling and Simulation*, pp. 219–225, 2010.

[82] A. Konstantinidis, C. Charalambous, A. Zhou, and Q. Zhang, "Multi-objective mobile agent-based sensor network routing using MOEA/D," in *IEEE Congress on Evolutionary Computation*, pp. 1–8, 2010.

[83] P. A. N. Bosman and D. Thierens, "Multi-objective optimization with diversity preserving mixture-based iterated density estimation evolutionary algorithms," *International Journal of Approximate Reasoning*, vol. 31, no. 3, pp. 259–289, 2002.

[84] M. Laumanns and J. Ocenasek, "Bayesian optimization algorithms for multi-objective optimization," in *Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature*, pp. 298–307, 2002.

[85] M. Costa and E. Minisci, "MOPED: A multi-objective Parzen-based estimation of distribution algorithm for continuous problems," in *Proceedings of the Second International Conference on Evolutionary Multi-criterion Optimization*, pp. 282–294, 2003.

[86] H. Li, Q. Zhang, E. Tsang, and J. A. Ford, "Hybrid estimation of distribution algorithm for multiobjective knapsack problem," in *Fourth European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 145–154, 2004.

[87] T. Okabe, Y. Jin, B. Sendhoff, and M. Olhofer, "Voronoi-based estimation of distribution algorithm for multi-objective optimization," in *IEEE Congress on Evolutionary Computation*, pp. 1594–1601, 2004.

[88] M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective hBOA, clustering, and scalability," in *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pp. 663–670, 2005.

[89] X. Zhong and W. Li, "A decision-tree-based multi-objective estimation of distribution algorithm," in *Proceedings of the 2007 International Conference on Computational Intelligence and Security*, pp. 114–118, 2007.

[90] L. Martí, J. García, A. Berlanga, and J. M. Molina, "Solving complex high-dimensional problems with the multi-objective neural estimation of distribution algorithm," in *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pp. 619–626, 2009.

[91] H. Mühlenbein, "The equation for response to selection and its use for prediction," *Evolutionary Computation*, vol. 5, no. 3, pp. 303–346, 1998.

[92] A. W. Iorio and X. Li, "Solving rotated multiobjective optimization problems using differential evolution," in *Procdeeding of Artificial Intelligence: Advances in Artificial Intelligence*, pp. 861–872, 2004.

[93] J. D. Knowles, "PaeEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[94] E. Zitzler, T. L., M. Laumanns, C. M. Fonseca, and V. G. Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[95] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm test suites," *ACM Symposium on Applied Computing*, pp. 351–357, 1999.

[96] D. A. Van Veldhuizen and G. B. Lamont, "On measuring multiobjective evolutionary algortihm performance," in *IEEE Congress on Evolutionary Computation*, pp. 204–211, 2000.

[97] K. Deb, "Multi-objective genetic algorithm: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, 1999.

[98] D. A. Van Veldhuizen, *Multiobjective evolutinoary algorithm: Classification, analyzes, and new innovations*. Ph.D. Dissertation, Air Force Institute of Technology , Wright-Patterson AFB, 1999.

[99] T. Okabe, Y. Jin, M. Olhofer, and B. Sendhoff, "On test functions for evolutionary multi-objective optimization," in *Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature*, pp. 792–802, 2004.

[100] V. L. Huang, A. K. Qin, K. Deb, E. Zitzler, P. N. Suganthan, J. J. Liang, M. Preuss, and S. Huband, "Problem definitions for performance assessment of multi-objective optimizatio algorithms," Technical Report, Nanyang Technological University, Singapore, 2007.

[101] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[102] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multi-objective optimization," KanGAL Report 2001001, Kanpur Genetic Algorithms Lab, Indian Institute of Technology, 2001.

[103] Q. Zhang, A. Zhou, P. N. Zhao, S. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," Technical Report CES-487, The School of Computer Science and Electronic Engineering, 2009.

[104] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[105] M. Pelikan and D. E. Goldberg, "Hierarchical BOA solves ising spin glasses and MAXSAT," in *Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation*, pp. 1271–1282, 2003.

[106] T. K. Paul and H. Iba, "Reinforcement learning estimation of distribution algorithm," in *Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation*, pp. 1259–1270, 2003.

[107] S. K. Shakya, *DEUME : A framework for an estimation of distribution algorithm based on markov random fields*. Ph.D. Dissertation, Robert Gordon University, Aberdeen, Scotland, 2006.

[108] Q. Zhang, "On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 1, pp. 80–93, 2004.

[109] H. Lu and G. G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 4, pp. 325–343, 2003.

[110] K. Deb, A. Sinha, and S. Kukkonen, "Multi-objective test problems, linkages, and evolutionary methodologies," in *Proceedings of the Eighth Annual Conference on Genetic and Evolutionary Computation*, pp. 1141–1148, 2006.

[111] O. Schutze, A. Lara, and C. A. Coello Coello, "On the influence of the number of objectives on the hardness of a multiobjective optimization problem," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 444–455, 2011.

[112] Y. Mei, K. Tang, and X. Yao, "Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 2, pp. 151–165, 2003.

[113] G. G. Yen and W. F. Leong, "A multiobjective particle swarm optimizer for constrained optimization," *International Journal of Swarm Intelligence Research*, vol. 2, no. 1, pp. 1–23, 2011.

[114] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *Proceedings of the 24th International Conference on Machine Learning*, pp. 791–798, 2007.

[115] T. Tieleman, "Training restricted Boltzmann machines using approximations to the likelihood gradient," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 1064–1071, 2008.

[116] M. Yasuda and K. Tanaka, "Approximate learning algorithm for restricted Boltzmann machines," in *Proceedings of the 2008 International Conference on Computational Intelligence for Modelling Control & Automation*, pp. 692–697, 2008.

[117] K. Sastry, D. E. Goldberg, and M. Pelikan, "Limits of scalability of multiobjective estimation of distribution algorithms," in *IEEE Congress on Evolutionary Computation*, pp. 2217–2224, 2005.

[118] H. Soh and M. Kirley, "moPGA: Towards a new generation of multi-objective genetic algorithms," in *IEEE Congress on Evolutionary Computation*, pp. 1702–1709, 2006.

[119] G. E. Hinton, "Training products of experts by minimizing constrative divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[120] G. E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[121] P. V. Gehler, A. D. Holub, and M. Welling, "The rate adapting possion model for information retrieval and object recognition," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 337–344, 2006.

[122] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning," in *Artificial Intelligence and Statistics*, pp. 17–22, Society for Artificial Intelligence and Statistics, 2005.

[123] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *IEEE Congress on Evolutionary Computation*, pp. 825–830, 2002.

[124] T. Chen, K. Tang, G. Chen, and X. Yao, "Analysis of computational time of simple estimation of distribution algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 1–22, 2009.

[125] P. A. N. Bosman and D. Thierens, "Continuous iterated density estimation evolutionary algorithm within the IDEA framework," in *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference*, pp. 197–200, 2000.

[126] M. Pelikan, D. E. Goldberg, and F. G. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5–20, 2002.

[127] A. W. Iorio and X. Li, "Rotated test problems for assessing the performance of multi-objective optimization algorithms," in *Proceedings of the eighth annual conference on Genetic and evolutionary computation*, pp. 683–690, 2006.

[128] L. R. Emmendorfer and A. Pozo, "Effective linkage learning using low-order statistics and clustering," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 6, pp. 1233–1246, 2009.

[129] M. D. Platel, S. Schliebs, and N. Kasabov, "Quantum-inspired evolutionary algorithm: A multimodel EDA," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 6, pp. 1218–1232, 2009.

[130] L. Qiang and Y. Xin, "Clustering and learning gaussian distribution for continuous optimization," *IEEE Transactions on Systems, Man and Cybernetics: Part C (Applications and Reviews)*, vol. 32, no. 2, pp. 195–204, 2005.

[131] E. S. Correa and J. L. Shapiro, "Model complexity vs. performance in the Bayesian optimization algorithm," in *Proceedings of the ninth international conference on Parallel Problem Solving from Nature*, pp. 998–1007, 2006.

[132] C. F. Lima, M. Pelikan, D. E. Goldberg, F. G. Lobo, K. Sastry, and M. Hauschild, "Influence of selection and replacement strategies on linkage learning in BOA," in *IEEE Congress on Evolutionary Computation*, pp. 1083–1090, 2007.

[133] H. Wu and J. L. Shapiro, "Does over-fitting affect performance in estimation of distribution algorithms," in *Proceedings of the eighth annual conference on Genetic and evolutionary computation*, pp. 433–434, 2006.

[134] M. Hauschild, M. Pelikan, K. Sastry, and C. Lima, "Analyzing probabilistic models in hierarchical BOA," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 6, pp. 1199–1217, 2009.

[135] C. Poloni, "Hybrid GA for multi-objective aerodynamic shape optimization," in *G. Winter, J. Periaux, M. Galan et al. (eds.) Genetic algorithms in engineering and computer science. Wiley, Chichester*, pp. 397–414, 1997.

[136] N. Le Roux and Y. Bengio, "Representational power of restricted boltzmann machines and deep belief networks," *Neural Computation*, vol. 20, no. 6, pp. 1631–1649, 2008.

[137] F. Kursawe, "A variant of evolution strategies for vector optimization," in *Proceedings of the First Workshop on Parallel Problem Solving from Nature*, pp. 193–197, 1991.

[138] C. K. Goh, Y. S. Ong, K. C. Tan, and E. J. Teoh, "An investigation on evolutionary gradient search for multi-objective optimization," in *IEEE Congress on Evolutionary Computation*, pp. 3741–3746, 2008.

[139] D. S. Liu, K. C. Tan, C. K. Goh, and W. K. Ho, "A multiobjective memetic algorithm based on particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Part B (Cybernetics)*, vol. 37, no. 1, pp. 42–50, 2007.

[140] E. Zitzler, T. L., and J. Bader, "On set-based multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 58–79, 2010.

[141] K. Deb and A. Agrawal, "Understanding interactions among genetic algorithm parameters," in *Proceedings of the Fifth Workshop on Foundations of Genetic Algorithms*, pp. 265–286, 1998.

[142] D. Goldberg, K. Deb, and B. Korb, "Messy genetic algorithms: Motivation, analysis, and first results," *Complex Systems*, vol. 3, no. 5, pp. 493–530, 1989.

[143] G. R. Harik, *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. Ph.D. Dissertation, University of Michigan, Ann Arbor, MI, USA, 1997.

[144] H. G. Beyer, "Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 239–267, 2000.

[145] P. Darwen and J. Pollack, "Coevolutionary learning on noisy tasks," in *IEEE Congress on Evolutionary Computation*, pp. 1724–1731, 1999.

[146] E. J. Hughes, "Evolutionary multi-objective ranking with uncertainty and noise," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, pp. 329–343, 2001.

[147] J. Teich, "Pareto-front exploration with uncertain objective," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, pp. 314–328, 2001.

[148] L. T. Bui, H. A. Abbass, and D. Essam, "Fitness inheritance for noisy evolutionary multi-objective optimization," in *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pp. 779–785, 2005.

[149] P. Limbourg, "Multi-objective optimization of problems with epistemic uncertainty," in *Proceedings of the Third International Conference on Evolutionary Multi-criterion Optimization*, pp. 413–427, 2005.

[150] Y. Hong, Q. Ren, J. Zeng, and Y. Chang, "Convergence of estimation of distribution algorithms in optimization of additively noisy fitness functions," in *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence*, pp. 219–223, 2005.

[151] Y. Hong, Q. Ren, and J. Zeng, "Optimization of noisy fitness functions with univariate marginal distribution algorithm," in *IEEE Congress on Evolutionary Computation*, pp. 1410–1417, 2005.

[152] J. Robinson and Y. Rahmat-Samii, "Particle swarm optimization in electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 2, pp. 397–407, 2004.

[153] M. Iqbal, A. A. Freitas, and C. G. Johnson, "Protein interaction inference using particle swarm optimization algorithm," in *Proceedings of the Sixth European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 61–70, 2008.

[154] F. Azevedo, Z. A. Vale, P. Oliveira, and H. Khodr, "A long-term risk management tool for electricity markets using swarm intelligence," *Electric Power Systems Research*, vol. 80, no. 4, pp. 380–389, 2010.

[155] D. Büche, P. Stoll, R. Dornberger, and P. Koumoutsakos, "Multiobjective evolutionary algorithm for the optimization of noisy combustion processes," *IEEE Transactions on Systems, Man and Cybernetics: Part C (Applications and Reviews)*, vol. 32, no. 4, pp. 460–473, 2002.

[156] L. T. Bui, H. A. Abbass, and D. Essam, "Localization for solving noisy multi-objective optimization problems," *Evolutionary Computation*, vol. 17, no. 3, pp. 379–409, 2009.

[157] M. Basseur, E. Zitzler, and E. Talbi, "A preliminary study on handling uncertainty in indicator-based multiobjective optimization," in *Proceedings of the 2006 International Conference on Applications of Evolutionary Computing*, pp. 727–739, 2006.

[158] H. Eskandari and C. D. Geiger, "Evolutionary multiobjective optimization in noisy problem environments," *Journal of Heuristics*, vol. 15, no. 6, pp. 559–595, 2009.

[159] P. Boonma and J. Suzuki, "A confidence-based dominance operator in evolutionary algorithms for noisy multiobjective optimization problems," in *Proceedings of the 2009 21st IEEE International Conference on Tools with Artificial Intelligence*, pp. 387–394, 2009.

[160] A. Syberfeldt, A. Ng, R. John, and P. Moore, "Evolutionary optimisation of noisy multiobjective problems using confidence-based dynamic resampling," *European Journal of Operational Research*, vol. 204, no. 3, pp. 533–544, 2010.

[161] B. L. Miller and D. E. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Systems*, vol. 9, no. 3, pp. 193–212, 1995.

[162] M. Chakraborty and U. Chakraborty, "An analysis of linear ranking and binary tournament selection in genetic algorithms," in *Proceedings of the First International Conference on Information Communications and Signal Processing*, pp. 407–411, 1997.

[163] A. P. Engelbrecht, *Fundamentals of computational swarm intelligence*. John Wiley and Sons, 2006.

[164] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4101–4108, 1997.

[165] W. Cedeno and D. K. Agrafiotis, "Application of niching particle swarms to QSAR and QSPR," in *Proceedings of the Fourteenth European Symposium on QSAR*, pp. 8–13, 2002.

[166] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[167] E. H. L. Aarts and J. H. M. Korst, "Boltzmann machines for travelling salesman problems," *European Journal of Operational Research*, vol. 39, no. 1, pp. 79–95, 1989.

[168] J. Blazewicz, M. Kasprzak, and W. Kuroczycki, "Hybrid genetic algorithm for dna sequencing with errors," *Journal of Heuristics*, vol. 8, no. 5, pp. 495–502, 2002.

[169] H. A. Eiselt and G. Laporte, "A combinatorial optimization problem arising in dartboard design," *Journal of the Operational Research Society*, vol. 42, no. 2, pp. 113–118, 1991.

[170] G. Reinelt, "Fast heuristics for large geometric traveling salesman problems," *INFORMS Journal on Computing*, vol. 4, no. 2, pp. 206–217, 1989.

[171] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 2, pp. 231–247, 1992.

[172] T. P. Bagchi, J. N. D. Gupta, and C. Sriskandarajah, "A review of TSP based approaches for flowshop scheduling," *European Journal of Operational Research*, vol. 169, no. 3, pp. 816–854, 2006.

[173] M. Jähne, X. Li, and J. Branke, "Evolutionary algorithms and multi-objectivization for the travelling salesman problem," in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pp. 595–602, 2009.

[174] M. Yang, L. Kang, and J. Guan, "An evolutionary algorithm for dynamic multi-objective TSP," in *Proceedings of the Second International Conference on Advances in Computation and Intelligence*, pp. 62–71, 2007.

[175] C. Y. Cheong, K. C. Tan, D. K. Liu, and C. J. Lin, "Multi-objective and prioritized berth allocation in container ports," *Annals of Operations Research*, vol. 180, no. 1, pp. 63–103, 2010.

[176] S. Y. Shin, I. H. Lee, and B. T. Zhang, "Multiobjective evolutionary optimization of dna sequences for reliable dna computing," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 2, pp. 143–158, 2005.

[177] M. Diaby, "The traveling salesman problem: A linear programming formulation," *Transactions On Mathematics*, vol. 6, no. 6, pp. 745–754, 2007.

[178] C. Malandraki and R. B. Dial, "A restricted dynamic programming heuristic algorithm for time dependent traveling salesman problem," *European Journal of Operational Research*, vol. 90, no. 1, pp. 45–55, 1996.

[179] M. Padberg and G. Rinaldi, "Branch-and-cut approach to a variant of the traveling salesman problem," *Journal of Guidance, Control, and Dynamics*, vol. 11, no. 5, pp. 436–440, 1988.

[180] J. B. Wu, S. W. Xiong, and N. Xu, "Simulated annealing algorithm based on controllable temperature for solving TSP," *Application Research of Computers*, vol. 24, no. 5, pp. 66–89, 2007.

[181] Y. Wu and X. Zhou, "Meliorative tabu search algorithm for TSP problem," *Journal of Computer Engineering and Applications*, vol. 44, no. 1, pp. 57–59, 2008.

[182] X. S. Yan, H. M. Liu, J. Yan, and Q. H. Wu, "A fast evolutionary algorithm for traveling salesman problem," in *Proceedings of the Third International Conference on Natural Computation*, pp. 85–90, 2007.

[183] M. Li, Z. Yi, and M. Zhu, "Solving TSP by using Lotka-Volterra neural networks," *Neurocomputing*, vol. 72, no. 16-18, pp. 3873–3880, 2009.

[184] J. Q. Yang, J. G. Yang, and G. L. Chen, "Solving large scale TSP using adaptive clustering method," in *Proceedings of the Second International Symposium on Computational Intelligence and Design*, pp. 44–51, 2009.

[185] L. Shi and Z. Li, "An improved Pareto genetic algorithm for multi-objective tsp," in *Proceedings of the Fifth International Conference on Natural Computation*, pp. 585–588, 2009.

[186] O. Yugay, I. KIm, B. Kim, and F. I. S. Ko, "Hybrid genetic algorithm for solving traveling salesman problem with sorted population," in *Proceedings of the Third International Conference on Convergence and Hybrid Information Technology*, pp. 1024–1028, 2008.

[187] G. Zhou and L. Jia, "A novel evolutionary algorithm for bi-objective symmetric traveling salesman problem," *Journal of Computational Information Systems*, vol. 4, no. 5, pp. 2051–2056, 2008.

[188] S. Elaoud, J. Teghem, and T. Louki1, "Multiple crossover genetic algorithm for the multiobjective traveling salesman problem," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 939–946, 2010.

[189] C. Gonzales, *Contributions on theoretical aspects of estimation of distribution algorithms*. Ph.D. Dissertation, University of the Basque Country, 2005.

[190] J. A. Lozano, P. Larrañaga, and E. Bengoetxea, *Towards a new evolutionary computation: Advances on estimation of distribution algorithms (Studies in Fuzziness and Soft Computing)*. Springer, 2006.

[191] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," Technical Report: CS-94-163, 1994.

[192] W. Peng, Q. Zhang, and H. Li, "Comparison between MOEA/D and NSGA-II on the multiobjective travelling salesman problem," *Studies in Computational Intelligence*, vol. 171, pp. 309–324, 2009.

[193] C. García-Martínez, O. Cordón, and F. Herrera, "An empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 61–72, 2004.

[194] F. Herrera, C. García-Martínez, and O. Cordón, "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," *European Journal of Operational Research*, vol. 80, no. 1, pp. 116–148, 2007.

[195] L. Manuel and S. Thomas, "The impact of design choices of multiobjective antcolony optimization algorithms on performance: An experimental study on the biobjective tsp," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pp. 71–78, 2010.

[196] S. Chen, M. Chen, P. Zhang, Q. Zhang, and Y. Chen, "Guidelines for developing effective estimation of distribution algorithms in solving single machine scheduling problems," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6441–6451, 2010.

[197] B. Jarboui, M. Eddaly, and P. Siarry, "An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems," *Computers and Operations Research*, vol. 36, no. 9, pp. 2638–2646, 2009.

[198] A. Salhi, J. A. V. Rodríguez, and Q. Zhang, "An estimation of distribution algorithm with guided mutation for a complex flow shop scheduling problem," in *Proceedings of the Ninth Annual Conference on Genetic and Evolutionary Computation*, pp. 570–576, 2007.

[199] L. Lin, "Maximum entropy estimation of distribution algorithm for jssp under uncertain information based on rough programming," *International Workshop on Intelligent Systems and Application*, pp. 1–4, 2009.

[200] C. Xu, J. Xu, and H. Chang, "Ant colony optimization based on estimation of distribution for the traveling salesman problem," in *Proceedings of the Fifth International Conference on Natural Computation*, pp. 19–23, 2009.

[201] M. A. Villalobos-arias, G. T. Pulido, and C. A. Coello Coello, "A proposal to use stripes to maintain diversity in a multi-objective particle swarm optimizer," in *Proceedings 2005 IEEE Swarm Intelligence Symposium*, pp. 22–29, 2005.

[202] D. A. Van Veldhuizen and G. B. Lamont, "Evolutionary computation and convergence to a Pareto front," in *Late Breaking Papers at the Genetic Programming 1998 Conference*, pp. 221–228, 1998.

[203] R. D. Angle, W. L. Caudle, R. Noonon, and A. Whinston, "Computer assisted school bus scheduling," *Management Science*, vol. 18, no. 6, pp. 278–288, 1972.

[204] H. A. Saleh and R. Chelouah, "The design of the navigation satellite system surveying networks using genetic algorithms," *Engineering Application of Artificial Intelligence*, vol. 17, no. 1, pp. 111–122, 2004.

[205] K. C. Gilbert and R. B. Hofstra, "A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem," *Decision Science*, vol. 23, no. 1, pp. 250–259, 1992.

[206] L. Tang, J. Liu, A. Rong, and Z. Yang, "A multiple traveling salesman problem model for hot rolling scheduling in shanghai baoshan iron and steel complex," *European Journal of Operational Research*, vol. 124, no. 2, pp. 267–282, 2000.

[207] Y. Zhong, J. Liang, G. Gu, R. Zhang, and H. Yang, "An implementation of evolutionary computation for path planning of cooperative mobile robots," in *Proceedings of the Fourth World Congress on Intelligent Control and Automation*, vol. 3, pp. 1798–1802, 2002.

[208] A. Singh and A. S. Baghel, "A new grouping genetic algorithm approach to the multiple traveling salesperson problem," *Soft Computing*, vol. 13, no. 1, pp. 95–101, 2009.

[209] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *The International Journal of Management Science*, vol. 34, no. 3, pp. 209–219, 2005.

[210] T. Zhang, W. A. Gruver, and M. H. Smith, "Team scheduling by genetic search," in *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials*, pp. 829–844, 1999.

[211] Y. B. Park, "A hybrid genetic algorithm for the vehicle scheduling problem with due times and times deadlines," *International Journal of Production Economics*, vol. 73, no. 2, pp. 175–188, 2001.

[212] A. E. Carter and C. T. Ragsdale, "A new approach to solving the multiple traveling salesperson problem using genetic algorithms," *European Journal of Operational Research*, vol. 175, no. 1, pp. 246–257, 2006.

[213] F. Zhao, J. Dong, S. Li, and X. Yang, "An improved genetic algorithm for multiple traveling salesman problem," in *Proceedings of the Second International Asia Conference on Informatics in Control, Automation and Robotics*, pp. 493–495, 2008.

[214] A. Király and J. Abonyi, "Optimization of multiple traveling salesman problem by a novel representation based genetic algorithm," in *Intelligent Computational Optimization in Engineering*, vol. 366, pp. 241–269, Springer, 2010.

[215] Y. S. Ong, M. Lim, and X. S. Chen, "Memetic computation - past, present & future," *IEEE Computational Intelligence Magazine*, vol. 5, no. 2, pp. 24–31, 2010.

[216] J. Y. Chia, C. K. Goh, K. C. Tan, and V. A. Shim, "Memetic informed evolutionary optimization via data dining," *Memetic Computing*, vol. 3, no. 2, pp. 73–88, 2011.

[217] R. Solomon, "Evolutionary algorithms and gradient search: Similarities and differences," *IEEE Transaction on Evolutionary Computation*, vol. 2, no. 2, pp. 45–55, 1998.

[218] W. T. Koo, C. K. Goh, and K. C. Tan, "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment," *Memetic Computing*, vol. 2, no. 2, pp. 87–110, 2010.

[219] C. Okonjo, "An effective method of balancing the workload amongst salesma," *Omega*, vol. 16, no. 2, pp. 159–163, 1998.

[220] H. A. Abbass and R. Sarker, "The Pareto differential evolution algorithm," *International Journal of Artificial Intelligence Tools*, vol. 11, no. 4, pp. 531–552, 2002.

[221] S. Kukkonen and J. Lampinen, "GDE3: The third evolution step of generalized differential evolution," in *IEEE Congress on Evolutionary Computation*, pp. 443–450, 2005.

[222] K. Deb and T. Goel, "Multi-objective evolutionary algorithms for engineering shape design," *International Series in Operations Research and Management Science*, vol. 48, pp. 147–175, 2003.

[223] J. D. Knowles and D. Corne, "Memetic algorithms for multiobjective optimization: Issues, methods and prospects," in *Recent Advances in Memetic Algorithms*, vol. 166, pp. 313–352, Studies in Fuzziness and Soft Computing, 2005.

[224] J. D. Knowles and D. W. Corne, "M-PAES: A memetic algorithm for multiobjective optimization," in *IEEE Congress on Evolutionary Computation*, pp. 325–332, 2000.

[225] T. Murata and H. Ishibuchi, "MOGA: Multi-objective genetic algorithms," in *IEEE Congress on Evolutionary Computation*, pp. 289–294, 1995.

[226] M. Caramia and P. Dell'Olmo, *Multi-objective management in freight logistics: Increasing capacity, service level and safety with optimization algorithms*. Springer, 2008.

[227] M. Emmerich, N. Beume, and B. Naujoks, "An EMO algorithm using the hypervolume measure as selection criterion," in *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization*, pp. 62–76, 2005.

Appendix A

Performance Metrics

Four performance metrics that are applied in this thesis are illustrated in this section. For description purposes, PF^* is a set of evolved solutions and PF is the set of Pareto optimal solutions. The definitions of the indicators are presented below.

1. **Generational Distance (GD):** Generational distance (GD) [202] is a unary performance indicator which is defined as

$$GD = \sqrt{\frac{\sum_{i=1}^{N} d(p^*, p)_i^2}{N}}$$

where N is the number of solutions in PF^*, $p \in PF$, $p^* \in PF^*$, and $d(p^*, p)_i$ is the minimum Euclidean distance in the objective space between p^* and p for each member i. GD illustrates the convergence ability of the algorithm by measuring the closeness between the Pareto optimal front and the evolved Pareto front. Thus, a lower value of GD shows that the evolved Pareto front is closer to the Pareto optimal front. This indicator is a representative metric which provides a quantitative measurement for the proximity goal of multi-objective optimization.

2. **Maximum Spread (MS):** Maximum spread (MS) is a unary performance indicator which measures how well the PF is covered by PF^*. The mathematical formulation of this metric is shown below:

$$MS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[\frac{min(f_i^{max} - F_i^{max}) - max(f_i^{min} - F_i^{min})}{F_i^{max} - F_i^{min}}\right]^2}$$

where F_i^{max} and F_i^{min} are the maximum and minimum of the i^{th} objective in PF, respectively, and f_i^{max} and f_i^{min} are the maximum and minimum of the i^{th} objective in PF^*, respectively [101]. A higher value of MS indicates that the evolved Pareto front has better spreading.

3. **Inverted Generational Distance (IGD):** Inverted generational distance (IGD) is a unary indicator which performs a near-similar calculation as done by GD [21,201]. The difference is that GD calculates the distance of each solution in PF^* to PF while IGD calculates the distance of each solution in PF to PF^*. In this indicator, both convergence and diversity are taken into consideration. A lower value of IGD implies that the algorithm has better optimization performance.

4. **Non-dominated Ratio (NR):** NR is an n-ary Pareto dominance metric proposed in [39] to compare the quality of solution sets from various algorithms. Representing the Pareto fronts evolved by n algorithms by PF_1^*, PF_2^* ,..., PF_n^*, this metric measures the non-dominated ratio of solutions in the Pareto front obtained by one algorithm compared to those obtained by the other algorithms. Mathematically, the NR is formulated as

$$NR(PF_1^*, PF_2^*, ..., PF_n^*) = \frac{|B \cap PF_1^*|}{B}$$

where, $B = \{b_i \mid \forall\, b_i \not\exists\, PF_j^* \in (PF_1^*, PF_2^*, ..., PF_n^*) \prec b_i\}$ and PF_1^* is the solution set under evaluation. This metric has been used in [138].

Appendix B

Multi-objective Test Problems

An MOP can be characterized in two main categories: fitness landscapes and Pareto optimal front geometries. In terms of fitness landscapes, an MOP may have a scalable number of objective functions. An MOP is difficult to solve in a higher number of objective functions since the selection pressure in selecting fitter individuals is reduced when problems consist of many conflicting objective functions (more than three). This is due to the high rate of non-dominance between individuals during the evolutionary process. This may hinder the search towards optimality or result in the population getting trapped in a local optimal. Besides, a huge fitness landscape may challenge an optimizer to search over the promising regions of the landscape. An MOP may also have a scalable number of decision variables. In problems with many decision variables, the complexity of the problems would increase with an increase in the number of variables. This is due to the enlargement of the search space and an increase in the number of possible moves towards optimality.

An MOP may also be characterized by modality. If an MOP has many local optima front, then it is a multimodal problem. If an MOP only consists of a single optimum front, then it is a unimodal problem. The multimodality of an MOP may cause optimizers to be trapped in any local optima solutions. A multimodal problem is more difficult to solve if it consists of deceptive optimum. In a deceptive MOP, the optimal front is placed in an unlikely place. Another characteristic of the fitness landscape is the mapping from the decision space to the objective space. If a set of evenly distributed samples are mapped to an unevenly distributed region of the objective space, then the problem is bias in nature. This may challenge an MOEA to generate a set of evenly distributed tradeoff solutions. An MOP may be separable or nonseparable. In separable problems, each decision variable can be optimized independently. On the other hand, nonseparable problems have certain level of dependencies between the decision variables.

In terms of the Pareto optimal front geometries, an MOP may have convex, concave, linear, disconnected, degenerate, and mixed geometries of Pareto optimal front. A Pareto optimal front is convex if the set of tradeoff solutions covers its convex hull. Similarly, a Pareto optimal front is concave if the set of tradeoff solutions covers its concave hull. A Pareto optimal front is linear if the set of tradeoff solutions is both concave and convex. An MOP has a degenerate Pareto front when the optimal front has a dimension lower than its objective space. A degenerate Pareto front may challenge an MOEA in generating a set of diverse tradeoff solutions. A Pareto optimal front may consist of several discontinuous subset of solutions. In other words, the Pareto optimal front is disconnected. Lastly, a mixed Pareto optimal front consists of several connected subsets with different geometries. A more detailed review, description, and analysis of the test problems can be referred to [104]. Table 1 lists the test problems together with their characteristics.

The ZDT test problems are extracted from [101].
ZDT1

$$f_1(\mathbf{x}) = x_1$$

$$f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} \right]$$

$$g(\mathbf{x}) = 1 + 9 \left(\frac{\sum_{i=2}^{n} x_i}{n-1} \right)$$

ZDT2

Same as ZDT1, except

$$f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \left(\frac{f_1(\mathbf{x})}{g(\mathbf{x})} \right)^2 \right]$$

ZDT3

Same as ZDT1, except

Table 1: Multi-objective test problems. S refers to scalable, m is the number of objective functions, K is a scalar parameter, n is the number of decision variables, SP refers to separable, NS refers to nonseparable, D refers to deceptive, U refers to unimodal, and M refers to multimodal.

Instance	m	n	Domain	Geometry	SP/NS	U/M	Bias
ZDT1	2	30(S)	$[0,1]^n$	Convex	SP	U	NO
ZDT2	2	30(S)	$[0,1]^n$	Concave	SP	U	NO
ZDT3	2	30(S)	$[0,1]^n$	Disconnected	SP	M	NO
ZDT4	2	30(S)	$[0,1]\times[-5,5]^{n-1}$	Convex	SP	M	NO
ZDT6	2	30(S)	$[0,1]^n$	Concave	SP	M	YES
DTLZ1	3(S)	$m+K-1$(S)	$[0,1]^n$	Linear	SP	M	NO
DTLZ2	3(S)	$m+K-1$(S)	$[0,1]^n$	Concave	SP	U	NO
DTLZ3	3(S)	$m+K-1$(S)	$[0,1]^n$	Concave	SP	M	NO
DTLZ4	3(S)	$m+K-1$(S)	$[0,1]^n$	Concave	SP	U	YES
DTLZ5	3(S)	$m+K-1$(S)	$[0,1]^n$	Degenerate	NS	U	NO
DTLZ6	3(S)	$m+K-1$(S)	$[0,1]^n$	Degenerate	NS	U	YES
DTLZ7	3(S)	$m+K-1$(S)	$[0,1]^n$	Disconnected	SP	M	NO
UF1	2	30(S)	$[0,1]\times[-1,1]^{n-1}$	Convex	SP	M	-
UF2	2	30(S)	$[0,1]\times[-1,1]^{n-1}$	Convex	NS	M	-
UF3	2	30(S)	$[0,1]^n$	Convex	NS	M	-
UF4	2	30(S)	$[0,1]\times[-2,2]^{n-1}$	Concave	NS	M	-
UF5	2	30(S)	$[0,1]\times[-1,1]^{n-1}$	Linear	NS	M	-
UF6	2	30(S)	$[0,1]\times[-1,1]^{n-1}$	Linear, Disconnected	NS	M	-
UF7	2	30(S)	$[0,1]\times[-1,1]^{n-1}$	Linear	NS	M	-
UF8	3	30(S)	$[0,1]^2\times[-2,2]^{n-2}$	Concave	SP	M	-
UF9	3	30(S)	$[0,1]^2\times[-2,2]^{n-2}$	Linear, Disconnected	SP	M	-
UF10	3	30(S)	$[0,1]^2\times[-2,2]^{n-2}$	Concave	NS	M	-
WFG1	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Convex, Mixed	SP	U	YES
WFG2	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Convex, Disconnected	NS	U	NO
WFG3	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Linear, Degenerate	NS	M	NO
WFG4	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	SP	M	NO
WFG5	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	SP	D	NO
WFG6	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	NS	U	NO
WFG7	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	SP	U	YES
WFG8	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	NS	U	YES
WFG9	2(S)	30(S)	$[0,2i]\ i\in\{1,...,n\}$	Concave	NS	M,D	YES

$$f_2(\mathbf{x}) = g(\mathbf{x})\left[1-\sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}}-\frac{f_1(\mathbf{x})}{g(\mathbf{x})}\text{sin}(10\pi x_1)\right]$$

ZDT4

Same as ZDT1, except

$$g(\mathbf{x}) = 1+10(n-1)+\sum_{i=2}^{n}\left[x_i^2-10\text{cos}(4\pi x_i)\right]$$

ZDT6

$$f_1(\mathbf{x}) = 1-exp(-4x_1)\text{sin}^6(6\pi x_1)$$

$$f_2(\mathbf{x}) = g(\mathbf{x})\left[1-\left(\frac{f_1(\mathbf{x})}{g(\mathbf{x})}\right)^2\right]$$

$$g(\mathbf{x}) = 1+9\left[\frac{\sum_{i=2}^{n}x_i}{n-1}\right]^{0.25}$$

The DTLZ test problems are extracted from [102].
DTLZ1

$$f_1(\mathbf{x}) = (1+g(\mathbf{x}))\,0.5\left(\prod_{i=1}^{m-1}x_i\right)$$

$$f_2(\mathbf{x}) = (1 + g(\mathbf{x}))\, 0.5 \left(\prod_{i=1}^{m-1} x_i \right) (1 - x_{m-1})$$

$$\vdots$$

$$f_m(\mathbf{x}) = (1 + g(\mathbf{x}))\, 0.5(1 - x_1)$$

$$g(\mathbf{x}) = 100 \left[(n - m + 1) + \sum_{i=m}^{n} \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right]$$

DTLZ2

$$f_1(\mathbf{x}) = (1 + g(\mathbf{x})) \left[\prod_{i=1}^{m-1} \cos(0.5\pi x_i) \right]$$

$$f_2(\mathbf{x}) = (1 + g(\mathbf{x})) \left[\prod_{i=1}^{m-2} \cos(0.5\pi x_i) \right] \sin(0.5\pi x_{m-1})$$

$$\vdots$$

$$f_m(\mathbf{x}) = (1 + g(\mathbf{x})) \sin(0.5\pi x_1)$$

$$g(\mathbf{x}) = \sum_{i=m}^{n} (x_i - 0.5)^2$$

DTLZ3

Same as DTLZ2, except

$$g(\mathbf{x}) = 100 \left[(n - m + 1) + \sum_{i=m}^{n} \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right]$$

DTLZ4

Same as DTLZ2, except

$$x_i = x_i^{\alpha}, \;\; i \in \{m, ..., n\}, \;\; \alpha = 10$$

DTLZ5

Same as DTLZ2, except

$$x_i = \frac{1 + 2g(\mathbf{x})x_i}{2(1 + g(\mathbf{x}))}, \;\; i \in \{2, ..., m - 1\}$$

DTLZ6

Same as DTLZ5, except

$$g(\mathbf{x}) = \sum_{i=m}^{n} x_i^{0.1}$$

DTLZ7

$$f_1(\mathbf{x}) = x_1$$

$$\vdots$$

$$f_{m-1}(\mathbf{x}) = x_{m-1}$$

$$f_m(\mathbf{x}) = (1+g(\mathbf{x}))\left[m - \sum_{i=1}^{m-1}\left(\frac{f_i(\mathbf{x})}{1+g(\mathbf{x})}(1+\sin(3\pi f_i(\mathbf{x})))\right)\right]$$

$$g(\mathbf{x}) = 1 + 9\left(\frac{\sum_{i=m}^{n} x_i}{n-m+1}\right)$$

The UF test problems are extracted from [103].
UF1

$$f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{i\in J_1}\left[x_i - \sin(6\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{i\in J_2}\left[x_i - \sin(6\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$J_1 = \{i|i \text{ is odd and } 2 \le i \le n\} \text{ and } J_2 = \{i|i \text{ is even and } 2 \le i \le n\}$$

UF2

$$f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{i\in J_1} y_i^2$$

$$f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{i\in J_2} y_i^2$$

$$J_1 = \{i|i \text{ is odd and } 2 \le i \le n\} \text{ and } J_2 = \{i|i \text{ is even and } 2 \le i \le n\}$$

$$y_i = \begin{cases} x_i - \left[0.3x_1^2\cos(24\pi x_1 + \frac{4i\pi}{n}) + 0.6x_1\right]\cos(6\pi x_1 + \frac{i\pi}{n}) & i \in J_1 \\ x_i - \left[0.3x_1^2\cos(24\pi x_1 + \frac{4i\pi}{n}) + 0.6x_1\right]\sin(6\pi x_1 + \frac{i\pi}{n}) & i \in J_2 \end{cases}$$

UF3

$$f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\left[4\sum_{i\in J_1} y_i^2 - 2\prod_{i\in J_1}\cos(\frac{20y_i\pi}{\sqrt{i}}) + 2\right]$$

$$f_2(\mathbf{x}) = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\left[4\sum_{i\in J_2} y_i^2 - 2\prod_{i\in J_2}\cos(\frac{20y_i\pi}{\sqrt{i}}) + 2\right]$$

$$J_1 = \{i|i \text{ is odd and } 2 \le i \le n\} \text{ and } J_2 = \{i|i \text{ is even and } 2 \le i \le n\}$$

$$y_i = x_i - x_1^{0.5\left(1.0+\frac{3(i-2)}{n-2}\right)},\ i \in \{2,...,n\}$$

UF4

$$f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{i\in J_1} h(y_i)$$

$$f_2(\mathbf{x}) = 1 - x_1^2 + \frac{2}{|J_2|}\sum_{i\in J_2} h(y_i)$$

$$J_1 = \{i|i \text{ is odd and } 2 \le i \le n\} \text{ and } J_2 = \{i|i \text{ is even and } 2 \le i \le n\}$$

$$y_i = x_i - \sin\left(6\pi x_1 + \frac{i\pi}{n}\right),\ i \in \{2,...,n\}$$

$$h(t) = \frac{|t|}{1+e^{2|t|}}$$

UF5

$$f_1(\mathbf{x}) = x_1 + \left(\frac{1}{2N} + \varepsilon\right)|\sin(2N\pi x_1)| + \frac{2}{|J_1|}\sum_{i\in J_1} h(y_i)$$

$$f_2(\mathbf{x}) = 1 - x_1 + \left(\frac{1}{2N} + \varepsilon\right)|\sin(2N\pi x_1)| + \frac{2}{|J_2|}\sum_{i\in J_2} h(y_i)$$

$J_1 = \{i | i \text{ is odd and } 2 \leq i \leq n\}$ and $J_2 = \{i | i \text{ is even and } 2 \leq i \leq n\}, N = 10$ and $\varepsilon = 0.1$

$$y_i = x_i - \sin(6\pi x_1 + \frac{i\pi}{n}),\ \ i \in \{2, ..., n\}$$

$$h(t) = 2t^2 - \cos(4\pi t) + 1$$

UF6

$$f_1(\mathbf{x}) = x_1 + \max\{0,\ 2(\frac{1}{2N} + \varepsilon)\sin(2N\pi x_1)\} + \frac{2}{|J_1|}\left[4\sum_{i\in J_1} y_i^2 - 2\prod_{i\in J_1}\cos(\frac{20y_i\pi}{\sqrt{i}}) + 2\right]$$

$$f_2(\mathbf{x}) = 1 - x_1 + \max\{0,\ 2(\frac{1}{2N} + \varepsilon)\sin(2N\pi x_1)\} + \frac{2}{|J_2|}\left[4\sum_{i\in J_2} y_i^2 - 2\prod_{i\in J_2}\cos(\frac{20y_i\pi}{\sqrt{i}}) + 2\right]$$

$J_1 = \{i | i \text{ is odd and } 2 \leq i \leq n\}$ and $J_2 = \{i | i \text{ is even and } 2 \leq i \leq n\}, N = 2$ and $\varepsilon = 0.1$

$$y_i = x_i - \sin\left(6\pi x_1 + \frac{i\pi}{n}\right),\ \ i \in \{2, ..., n\}$$

UF7

$$f_1(\mathbf{x}) = \sqrt[5]{x_1} + \frac{2}{|J_1|}\sum_{i\in J_1} y_i^2$$

$$f_2(\mathbf{x}) = 1 - \sqrt[5]{x_1} + \frac{2}{|J_2|}\sum_{i\in J_2} y_i^2$$

$J_1 = \{i | i \text{ is odd and } 2 \leq i \leq n\}$ and $J_2 = \{i | i \text{ is even and } 2 \leq i \leq n\}$

$$y_i = x_i - \sin\left(6\pi x_1 + \frac{i\pi}{n}\right),\ \ i \in \{2, ..., n\}$$

UF8

$$f_1(\mathbf{x}) = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|}\sum_{i\in J_1}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$f_2(\mathbf{x}) = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|}\sum_{i\in J_2}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$f_3(\mathbf{x}) = \sin(0.5x_1\pi) + \frac{2}{|J_3|}\sum_{i\in J_3}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$J_1 = \{i | 3 \leq i \leq n, \text{ and } i-1 \text{ is a multiplication of } 3\}$$

$$J_2 = \{i | 3 \leq i \leq n, \text{ and } i-2 \text{ is a multiplication of } 3\}$$

$$J_3 = \{i | 3 \leq i \leq n, \text{ and } i \text{ is a multiplication of } 3\}$$

UF9

$$f_1(\mathbf{x}) = 0.5\left[\max\{0,\ (1+\varepsilon)(1-4(2x_1-1)^2)\} + 2x_1\right]x_2 + \frac{2}{|J_1|}\sum_{i\in J_1}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$f_2(\mathbf{x}) = 0.5\left[\max\{0,\ (1+\varepsilon)(1-4(2x_1-1)^2)\} - 2x_1 + 2\right]x_2 + \frac{2}{|J_2|}\sum_{i\in J_2}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$$f_3(\mathbf{x}) = 1 - x_2 + \frac{2}{|J_3|}\sum_{i\in J_3}\left[x_i - 2x_2\sin(2\pi x_1 + \frac{i\pi}{n})\right]^2$$

$J_1,\ J_2,$ and J_3 are the same as $J_1,\ J_2,$ and J_3 from UF8, and $\varepsilon = 0.1$

UF10

$$f_1(\mathbf{x}) = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{|J_1|}\sum_{i\in J_1}\left[4y_i^2 - \cos(8\pi y_i) + 1\right]$$

$$f_2(\mathbf{x}) = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{|J_2|}\sum_{i\in J_2}\left[4y_i^2 - \cos(8\pi y_i) + 1\right]$$

$$f_2(\mathbf{x}) = \sin(0.5x_1\pi) + \frac{2}{|J_3|}\sum_{i\in J_3}\left[4y_i^2 - \cos(8\pi y_i) + 1\right]$$

$J_1,\ J_2,$ and J_3 are the same as $J_1,\ J_2,$ and J_3 from UF8

$$y_i = x_i - 2_x 2\sin\left(2\pi x_1 + \frac{i\pi}{n}\right),\ \ i \in \{3, ..., n\}$$

The WFG test problems are extracted from [104].
Common format of WFG test problems:
Given:

$$\mathbf{z} = \{z_1, ..., z_k, z_k + 1, ..., z_n\}$$

Minimize:

$$f_{j=1:m}(\mathbf{x}) = Dx_m + S_j h_j(x_1, ..., x_{m-1})$$

where

$$\begin{aligned}
\mathbf{x} &= \{x_1, ..., x_m\} \\
&= \{\max(t_m^p, A_1)(t_1^p - 0.5) + 0.5, ..., \max(t_m^p, A_{m-1})(t_{m-1}^p - 0.5) + 0.5, t_m^p\} \\
\mathbf{t^p} &= \{t_1^p, ..., t_m^p\} \leftarrow | \mathbf{t}^{p-1} \leftarrow | ... \leftarrow | \mathbf{t^1} \leftarrow | \mathbf{z}_{[\mathbf{0},\mathbf{1}]} \\
\mathbf{z}_{[\mathbf{0},\mathbf{1}]} &= \{z_{1,[0,1]}, ..., z_{n,[0,1]}\} \\
&= \{z_1/z_{1,\max}, ..., z_n/z_{n,\max}\}
\end{aligned}$$

Constants:

$S_{j=1:m} = 2m,\ \ D = 1,\ \ A_{1:m-1} = 1, k$ is the number of position-related parameters, and

l is the number of distance-related parameters

Shape functions ($h_{j=1:m}$):

Linear: represented by $\text{linear}_{1:m}(x_1, ..., x_{m-1})$
Convex: represented by $\text{convex}_{1:m}(x_1, ..., x_{m-1})$
Concave: represented by $\text{concave}_{1:m}(x_1, ..., x_{m-1})$
Mixed Convex and Concave: represented by $\text{mixed}_{1:m}(x_1, ..., x_{m-1})$
Disconnected: represented by $\text{disc}_{1:m}(x_1, ..., x_{m-1})$

Transformation functions:

Polynomial bias transformation: represented by b_poly(y, α)
Flat region bias transformation: represented by b_flat(y, A, B, C)
Parameter dependent bias transformation: represented by b_param$(y, \mathbf{y}^{'} A, B, C)$
Linear shift transformation: represented by s_linear(y, A)
Deceptive shift transformation: represented by s_decept(y, A, B, C)
Multi-modal shift transformation: represented by s_multi(y, A, B, C)
Weighted sum reduction transformation: represented by r_sum$(\mathbf{y}, \mathbf{w})$
Non-seperable reduction transformation: represented by r_nonsep$(\mathbf{y}, A)$

WFG1

$$
\begin{aligned}
h_{j=1:m-1} &= \text{convex}_j \\
h_m &= \text{mixed}_m \text{ (with } \alpha = 1 \text{ and } A = 5) \\
t^1_{i=1:k} &= y_i \\
t^1_{i=k+1:n} &= \text{s\_linear}(y_i, 0.35) \\
t^2_{i=1:k} &= y_i \\
t^2_{i=k+1:n} &= \text{b\_flat}(y_i, 0.8, 0.75, 0.85) \\
t^3_{i=1:n} &= \text{b\_poly}(y_i, 0.02) \\
t^4_{i=1:m-1} &= \text{r\_sum}(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \\
&\quad \{2[(i-1)k/(m-1)+1, ..., 2ik/(m-1)]\}) \\
t^4_m &= \text{r\_sum}(\{y_{k+1}, ..., y_n\}, \{2(k+1), ..., 2n\})
\end{aligned}
$$

WFG2

$$
\begin{aligned}
h_{j=1:m-1} &= \text{convex}_j \\
h_m &= \text{disc}_m \text{ (with } \alpha = \beta = 1 \text{ and } A = 5) \\
\mathbf{t}^1 &\text{ is the same as } \mathbf{t}^1 \text{ from WFG1 (linear shift)} \\
t^2_{i=1:k} &= y_i \\
t^2_{i=k+1:k+l/2} &= \text{r\_nonsep}(\{y_{k+2(i-k)-1}, y_{k+2(i-k)}\}, 2) \\
t^3_{i=1:m-1} &= \text{r\_sum}(\{y_{i-1k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\}) \\
t^3_m &= \text{r\_sum}(\{y_{k+1}, ..., y_{k+l/2}\}, \{1, ..., 1\})
\end{aligned}
$$

WFG3

$$
\begin{aligned}
h_{j=1:m} &= \text{linear}_j \\
\mathbf{t}^{1:3} &\text{ are the same as } \mathbf{t}^{1:3} \text{ from WFG2}
\end{aligned}
$$

WFG4

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
t^1_{1:n} &= \text{s\_multi}(y_i, 30, 10, 0.35) \\
t^2_{1:m-1} &= \text{r\_sum}(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\}) \\
t^2_m &= \text{r\_sum}(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})
\end{aligned}
$$

WFG5

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
t^1_{i=1:n} &= \text{s\_decept}(y_i, 0.35, 0.001, 0.05) \\
\mathbf{t}^2 &\text{ is the same as } \mathbf{t}^2 \text{ from WFG4}
\end{aligned}
$$

WFG6

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
\mathbf{t}^1 &\text{ is the same as } \mathbf{t}^1 \text{ from WFG1} \\
t^2_{i=1:m-1} &= \text{r\_nonsep}(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, k/(m-1)) \\
t^2_m &= \text{r\_nonsep}(\{y_{k+1}, ..., y_n\}, l)
\end{aligned}
$$

WFG7

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
t^1_{i=1:k} &= \text{b\_param}\left(y_i, \text{r\_sum}(\{y_{i+1}, ..., y_n\}, \{1, ..., 1\}), \frac{0.98}{49.98}, 0.02, 50\right) \\
t^1_{i=k+1:n} &= y_i \\
\mathbf{t}^2 &\text{ is the same as } \mathbf{t}^1 \text{ from WFG1} \\
\mathbf{t}^3 &\text{ is the same as } \mathbf{t}^2 \text{ from WFG4}
\end{aligned}
$$

WFG8

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
t^1_{i=1:k} &= y_i \\
t^1_{i=k+1:n} &= \text{b\_param}\left(y_i, \text{r\_sum}(\{y_i, ..., y_{i-1}\}, \{1, ..., 1\}), \frac{0.98}{49.98}, 0.02, 50\right) \\
\mathbf{t}^2 &\text{ is the same as } \mathbf{t}^1 \text{ from WFG1} \\
\mathbf{t}^3 &\text{ is the same as } \mathbf{t}^2 \text{ from WFG4}
\end{aligned}
$$

WFG9

$$
\begin{aligned}
h_{j=1:m} &= \text{concave}_j \\
t^1_{i=1:n-1} &= \text{b\_param}\left(y_i, \text{r\_sum}(\{y_{i+1}, ..., y_n\}, \{1, ..., 1\}), \frac{0.98}{49.98}, 0.02, 50\right) \\
t^1_n &= y_n \\
t^2_{i=1:k} &= \text{s\_decept}(y_i, 0.35, 0.001, 0.05) \\
t^2_{i=k+1:n} &= \text{s\_multi}(y_i, 30, 95, 0.35) \\
\mathbf{t}^3 &\text{ is the same as } \mathbf{t}^2 \text{ from WFG6}
\end{aligned}
$$